UNIVERSIDADE DA CORUÑA

Co-funded by the
Erasmus+ Programme
of the European Union

Introducción a la Inteligencia Artificial para educación pre-universitaria

Francisco Bellas

CFR Ferrol

Octubre 2021

Iniciación a la IA (1)

Obxectivos	Contidos	Criterios de avaliación	Estándares de aprendizaxe	Competencias clave
Bloque 1. Que é a Intelixencia Artificial				
• a • b • e • h • l	• B1.1 Que é a Intelixencia Artificial? o B1.2 Intelixencia natural fronte a Intelixencia Artificial o B1.3 Historia da Intelixencia Artificial o B1.4 Intelixencia Artificial forte fronte a Intelixencia Artificial feble	• B1.1 Coñecer á orixe da IA, a que campo de coñecemento pertence, a súa vinculación coa intelixencia humana e animal, e os dous principais enfoques da mesma.	• IIAB1.1.1 Define o significado de Intelixencia Artificial e sabe diferenciala da intelixencia natural. • IIAB1.1.2 Identifica o campo da Intelixencia Artificial dentro do campo de coñecemento adecuado (ciencias da computación) • IIAB1.1.3 Coñece a diferenza entre a Intelixencia Artificial forte e feble.	• CCL
• d • l • n	• B1.5 Elementos dun sistema intelixente o B1.6 Contornas reais, simuladas e virtuais o B1.7 Bloques básicos dun sistema de IA (percepción, representación, razoamento, aprendizaxe e actuación)	• B1.2 Coñecer os compoñentes básicos dun sistema de IA, entendendo que está situado nunha contorna real ou virtual coa que interactúa, e que a complexidade dos diferentes bloques pode variar.	• IIAB1.2.1 Identifica os elementos básicos dun sistema intelixente. • IIAB1.2.2 Distingue e define os diferentes tipos de contornas nos que pode estar situado un sistema intelixente	• CAA • CSIEE
• a • d • i	• B1.8 Campos de aplicación da Intelixencia Artificial	• B1.3 Coñecer os principais campos de aplicación real da IA (IA médica, robótica intelixente, contornas intelixentes: smart building, smart city, smart factory; sistemas de recomendación, videoxogos, chatbots, etc). e identificar os bloques básicos dun sistema intelixente en casos de uso concretos.	• IIAB1.3.1 Recoñece cando un sistema aplicado está baseado en IA ou non • IIAB1.3.2 Identifica os bloques básicos dun sistema intelixente en exemplos concretos de sistemas de IA en funcionamento	• CD • CAA • CCEC
Bloque 2. Áreas básicas da IA				
• c • f • g	• B2.1 Percepción e actuación en IA o B2.2 Sensorización contra percepción o B2.3 Sensores e actuadores básicos (distancia, orientación, luz, cor, motores, rodas, brazos) o B2.4 Sensores e percepción no ámbito da IA (cámaras e visión artificial, micrófonos e recoñecemento da fala, pantallas e interacción táctil) o B2.5 Actuadores e accións no ámbito da IA (altofalantes e produción de fala, navegación, manipulación, pantallas e outros interfaces virtuais) o B2.6 Interacción humano-máquina	• B2.1 Distinguir sensorización e percepción, coñecer os sensores e actuadores máis relevante na IA, coñecer a relevancia da interacción humano-máquina. Saber utilizar sensores e actuadores reais no ámbito da IA.	• IIAB2.1.1 Comprende a relevancia dos sensores e actuadores nos sistemas de IA, tanto reais como virtuais. • IIAB2.1.2 Distingue os sensores e actuadores propios dos sistemas intelixentes e por que proporcionan información de maior complexidade. • IIAB2.1.3 Coñece a relevancia da interacción humano-máquina na Intelixencia Artificial e comprende que todo sistema intelixente debe estar adaptado ás necesidades do público ao que vai dirixido.	• CD • CSIEE
• c • f • g	• B2.7 Aprendizaxe automática o B2.8 Conceptos básicos: preparación dos datos, aprendizaxe dos modelos e análise dos resultados o B2.9 Supervisado (clasificación e regresión) o B2.10 Non supervisado (agrupamento) o B2.11 Por reforzo (q-learning)	• B2.2 Coñecer os fundamentos da aprendizaxe automática, programación baseada nos datos, tratamento dos datos (conxuntos de adestramento e test), tipos de modelos básicos, análise de resultados. Comprender as diferencias entre os 3 tipos de aprendizaxe. Saber utilizar ferramentas básicas de aprendizaxe de modelos, e lograr un axuste de parámetros apropiado.	• IIAB2.2.1 Coñece que é o aprendizaxe automático e os seus fundamentos • IIAB2.2.2 Selecciona correctamente os datos para realizar o axuste dun modelo. • IIAB2.2.3 Utiliza adecuadamente ferramentas de aprendizaxe de modelos e logra analizar os resultados con rigor, comprendendo os factores que influencian o resultado.	• CMCCT • CSIEE

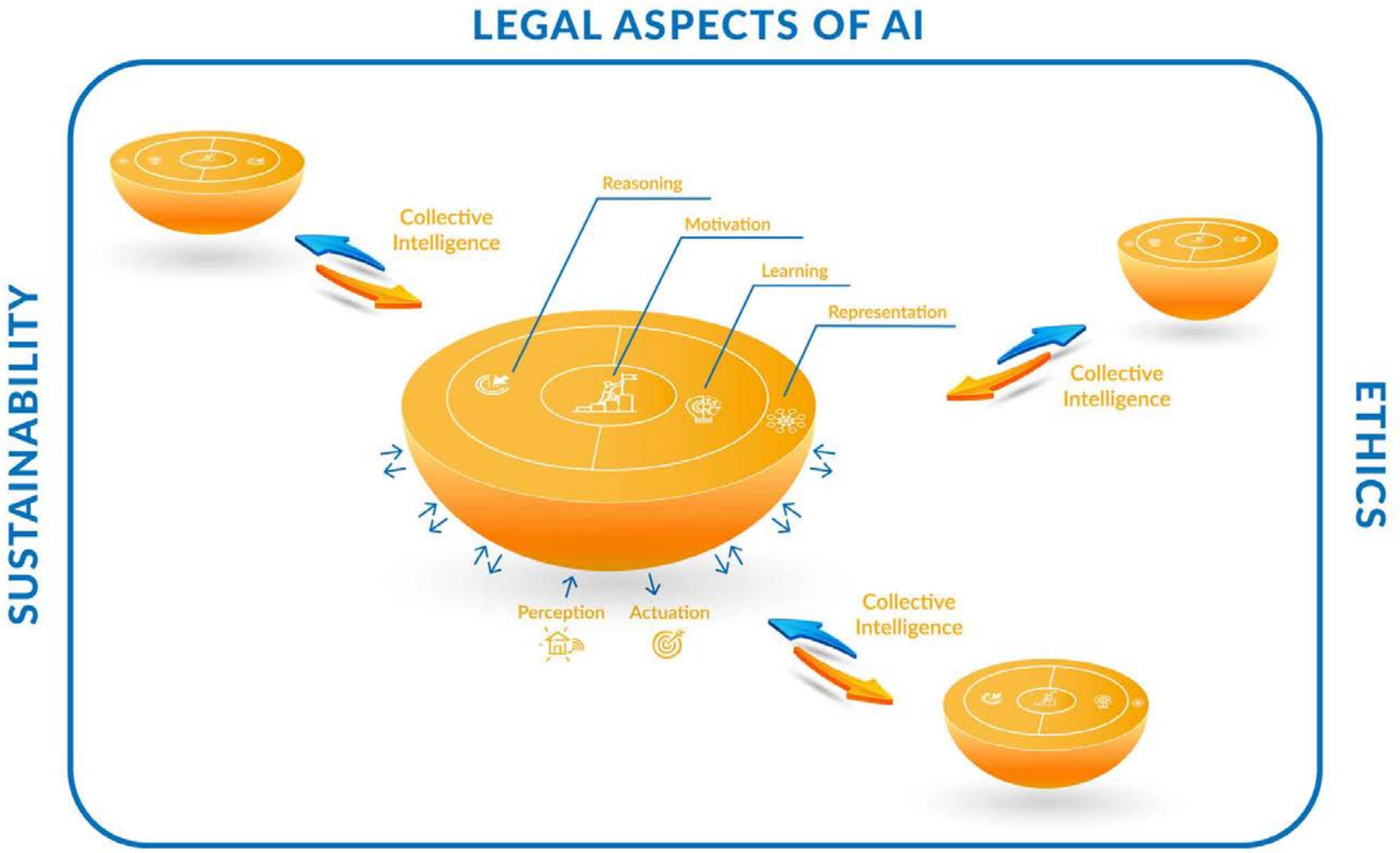
Iniciación a la IA (2)

Obxectivos	Contidos	Criterios de avaliación	Estándares de aprendizaxe	Competencias clave
			▪ IIAB2.2.4 Diferenza os tres tipos de aprendizaxe	
▪ c ▪ f ▪ g ▪ m	▪ B2.12 Representación e razoamento o B2.13 Como representar o coñecemento? o B2.14 Grafos e árbores de decisión o B2.15 Busca básica o B2.16 Fundamentos do razoamento probabilístico	▪ B2.3 Comprender como se representa computacionalmente o coñecemento a partir das percepcións, e como esta representación pode ser utilizada para os procesos de razoamento. Implementar programas que resolvan problemas sinxelos sobre árbores e grafos, utilizando algoritmos de busca sinxelos. Coñecer os fundamentos do razoamento probabilístico	▪ IIAB2.3.1 Comprende como se representan computacionalmente os datos e como se utiliza esta representación nos procesos de razoamento. ▪ IIAB2.3.2 Deseña árbores de decisión e grafos para resolver problemas sinxelos.	▪ CMCC ▪ CAA
▪ c ▪ f ▪ g ▪ m	▪ B2.17 IA colectiva o B2.18 Comunicación do coñecemento o B2.19 Contornas intelixentes	▪ B2.4 Coñecer as potencialidades da transmisión de información e coñecemento entre sistemas de IA. Comprender os fundamentos da IoT (Internet of Things) como base das contornas intelixentes: casas, edificios, cidades, fábricas.	▪ IIAB2.4.1 Comprende que os sistemas de IA futuros estarán interconectados formando parte dun ecosistema de IA colectiva (fontes de información e coñecemento distribuídas)	▪ CD
Bloque 3. Impacto da IA				
▪ a ▪ b ▪ e ▪ g	▪ B3.1 Ética da IA	▪ B3.1 Coñecer as consecuencias sociais do uso da IA en niveis como: a igualdade de raza e xénero, o desemprego, a toma de decisións morais e a influencia e desafío da privacidade que ten sobre os usuarios. Distinguir entre mitos e realidades da IA	▪ IIAB3.1.1 Identifica as consecuencias sociais do uso da IA e comprende as súas ventaxas e posibles riscos	▪ CCL ▪ CSC ▪ CCEC
▪ a ▪ b ▪ c ▪ d ▪ e	▪ B3.2 Aspectos legais da IA	▪ B3.2 Coñecer as implicacións legais do uso de sistemas autónomos e intelixentes	▪ IIAB3.2.1 Comprende as implicacións legais do uso de sistemas intelixentes, e identifica os posibles baleiros legais que existen sobre a IA dada a súa corta existencia.	▪ CAA ▪ CSC
▪ a ▪ b ▪ c ▪ p	▪ B3.3 Sostibilidade	▪ B3.3 Coñecer as consecuencias do crecemento de sistemas de IA na pegada do carbono, os residuos informáticos, o uso de redes de comunicacións. Coñecer a impacto positivo da IA nos Obxectivos de Desenvolvemento Sostible (ODS).	▪ IIAB3.3.1 Define o significado de sostibilidade e recoñece as consecuencias que trae o crecemento de sistemas de IA no relativo a este aspecto. Comprende os impactos positivos da IA nos ODS.	▪ CSC

Organización del curso

- Sesión 1: Introducción a la IA
- Sesión 2: Percepción y actuación en IA
- Sesión 3: Representación y razonamiento
- **Sesión 4: Aprendizaje automático**
- Sesión 5: IA colectiva
- Sesión 6: Impacto social de la IA

Temas de IA

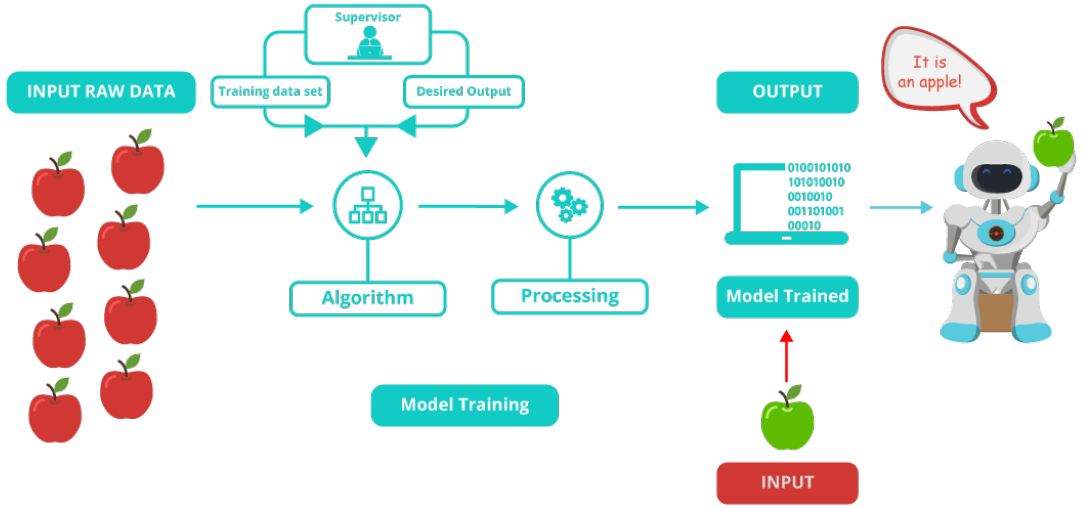

LEGAL ASPECTS OF AI
SUSTAINABILITY
ETHICS
Reasoning
Motivation
Learning
Representation
Collective Intelligence
Collective Intelligence
Collective Intelligence
Perception
Actuation

Contenidos sesión 4

- Aprendizaje automático
  - Principales métodos
  - Preparación de los datos
  - Análisis de resultados
  - Redes de neuronas artificiales
    - Perceptrón multicapa
    - Configuración
    - Criterios de parada

Definición

- El **aprendizaje automático** (machine learning - ML) es un campo de la inteligencia artificial en el cual se desarrollan algoritmos y modelos estadísticos que los sistemas computacionales utilizan para hacer predicciones o tomar decisiones **sin utilizar instrucciones explícitas, basándose en patrones e inferencias**.
  - En lugar de escribir el código de un programa, se alimenta con datos al algoritmo, y el este construye la lógica (el modelo) en base a dichos datos.

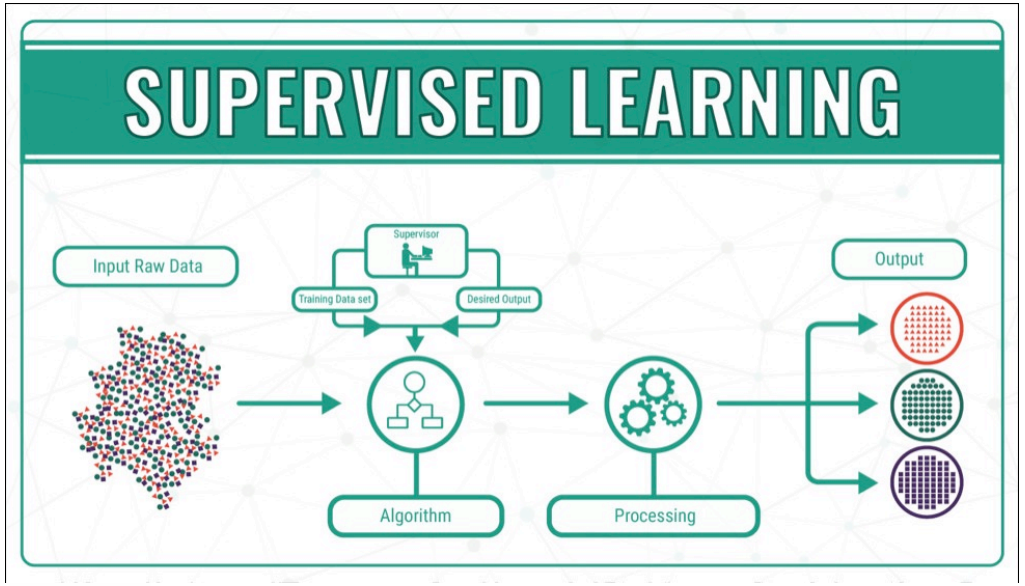


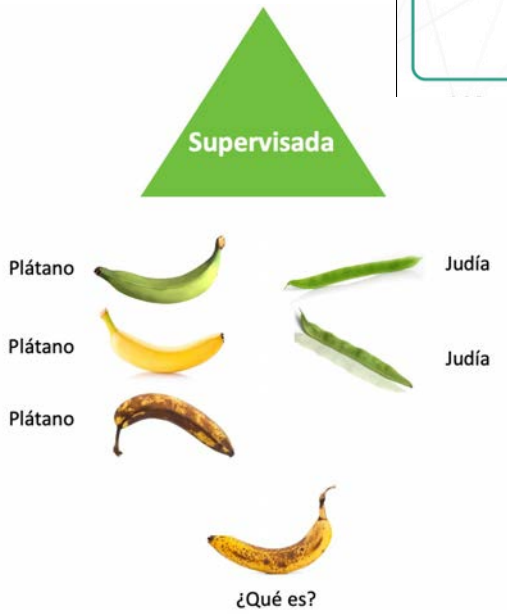
Fases

1. Obtención de los datos
   - Análisis del problema
2. Preparación de los datos
   - Minería de datos
3. Entrenamiento de los modelos
   - Conjunto de entrenamiento
4. Test de los modelos
   - Conjunto de test
5. Validación y mejora
   - Generalización del modelo

Principales métodos

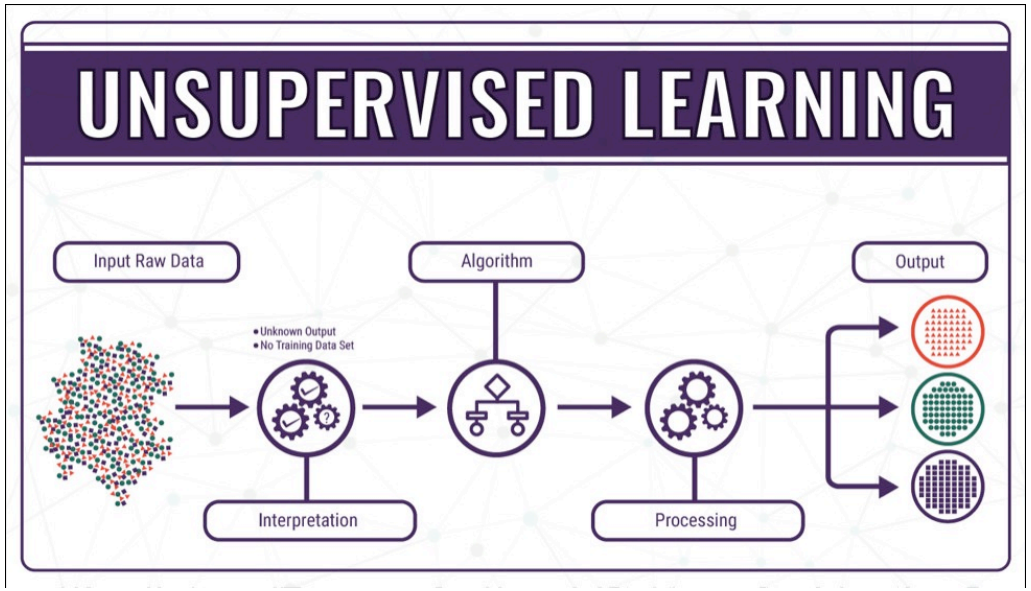
1. Obtención y preparación de los datos
   - **Etiquetar**
   - Seleccionar características
   - Normalizar
2. Selección del algoritmo
3. Ajuste de parámetros
4. Entrenamiento
5. Evaluación





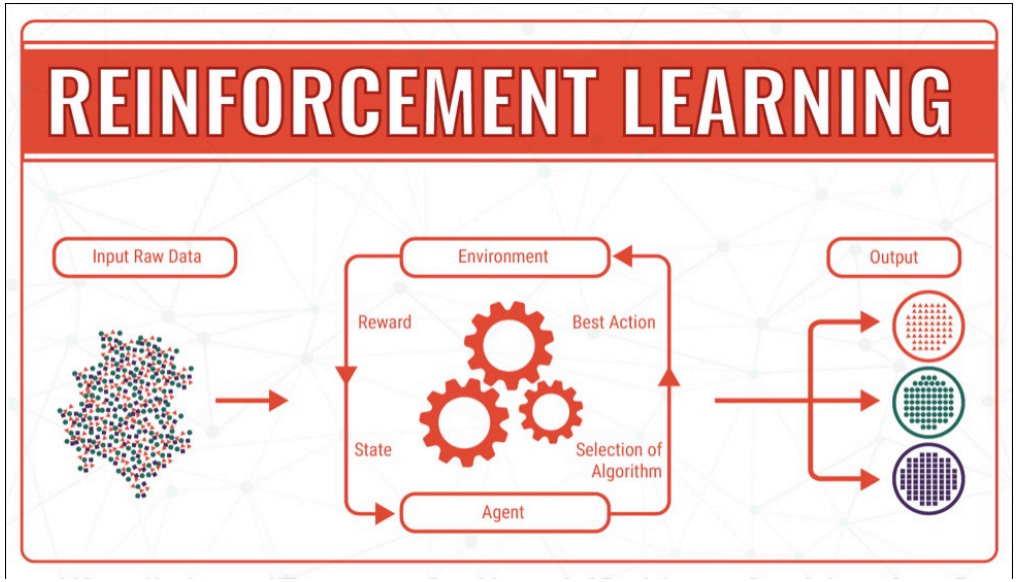
Principales métodos

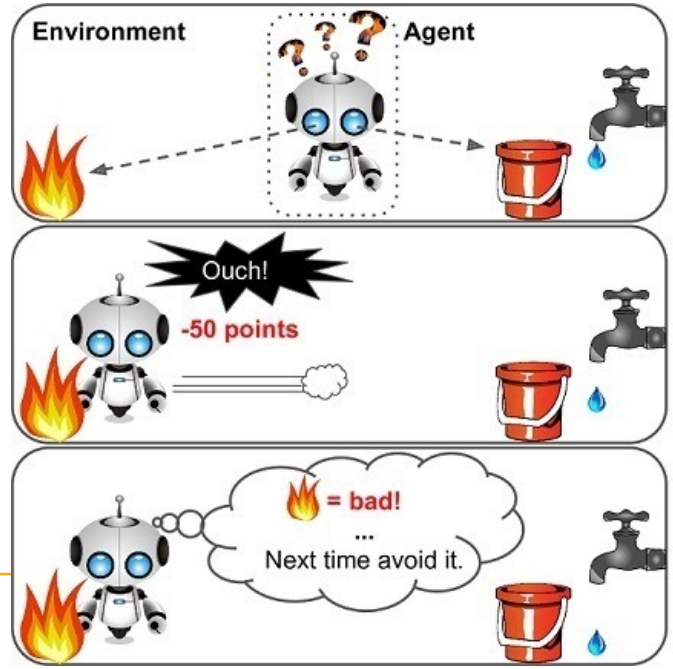
1. Obtención y preparación de los datos
   - Seleccionar características
   - Normalizar
2. Selección del algoritmo
3. Ajuste de parámetros
4. Entrenamiento
5. Evaluación



Principales métodos

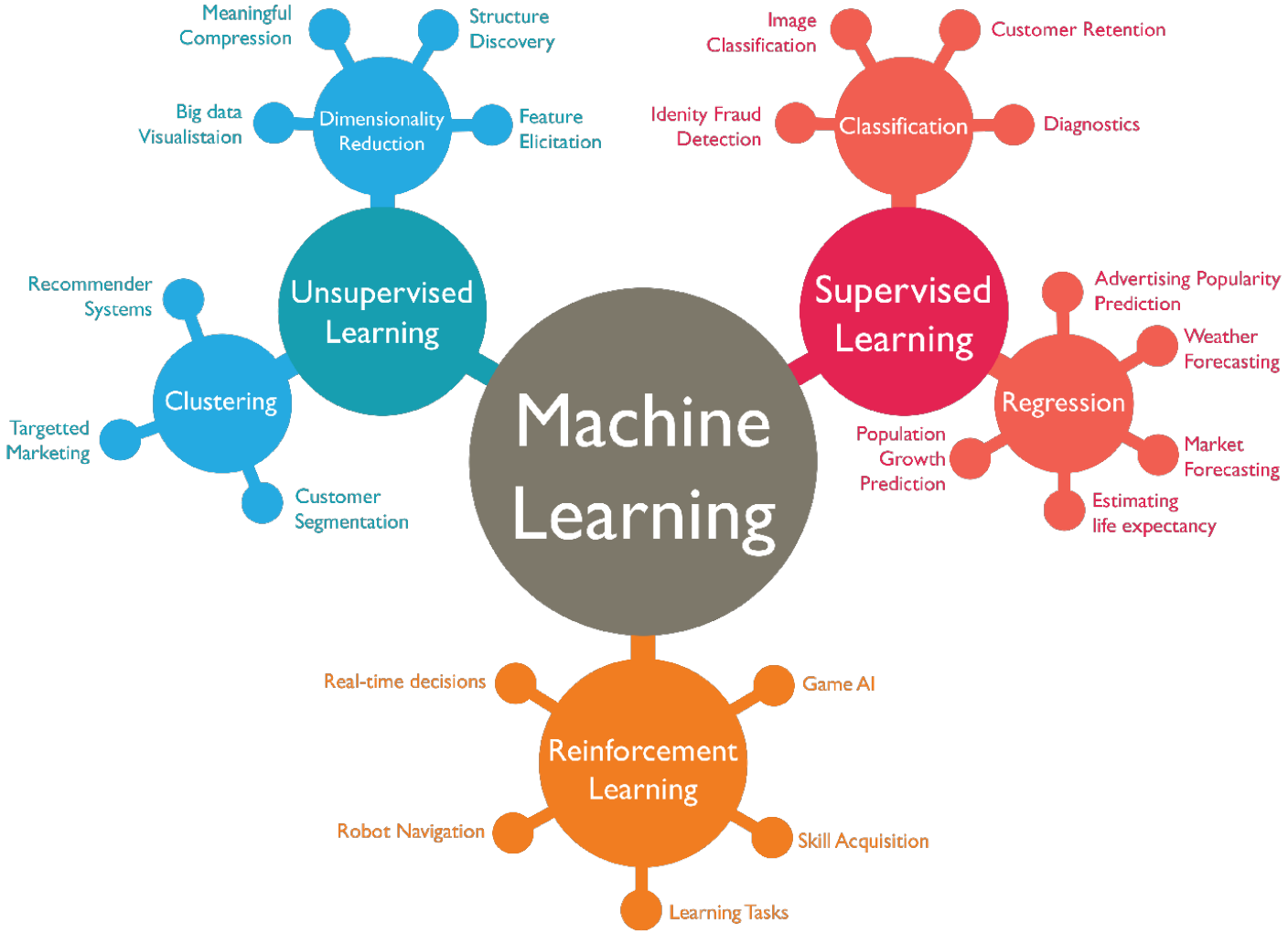
1. Obtención y preparación de los datos
   - **Tabla de recompensas**
   - Normalizar
2. Selección del algoritmo
3. Ajuste de parámetros
4. Entrenamiento
5. Evaluación



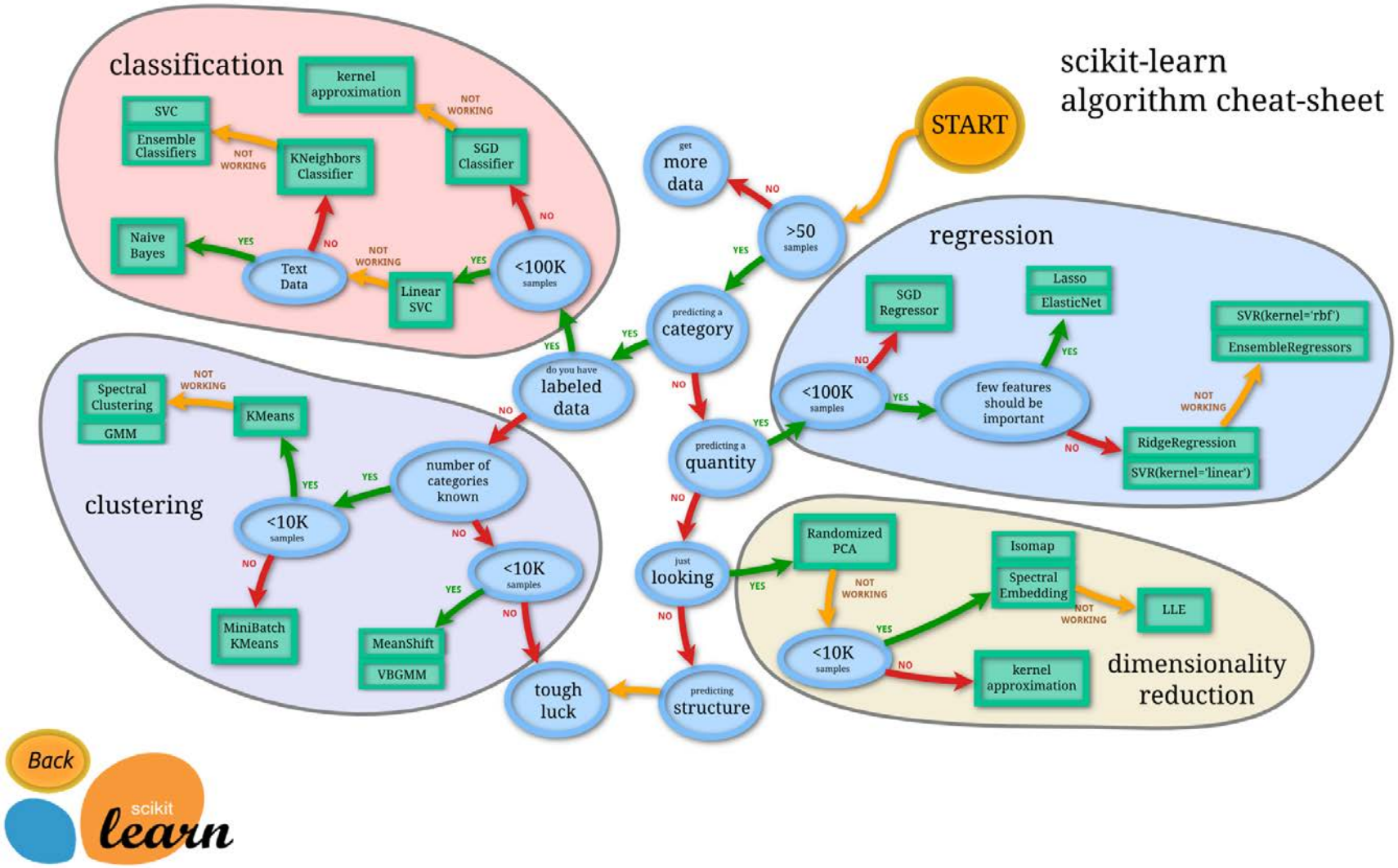


1. Observe
2. Select action using policy
3. Action!
4. Get reward or penalty
5. Update policy (learning step)
6. Iterate until an optimal policy is found

Principales métodos

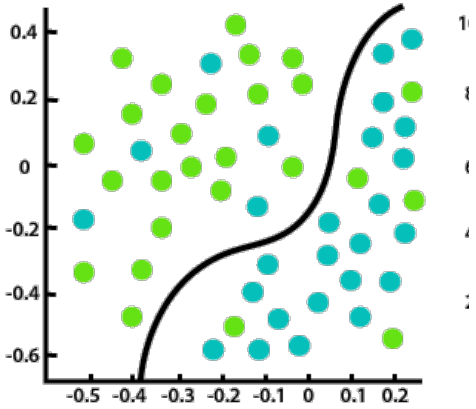


Algoritmos


scikit-learn
algorithm cheat-sheet
START
classification
kernel approximation
NOT WORKING
SVC
Ensemble Classifiers
NOT WORKING
KNeighbors Classifier
SGD Classifier
Naive Bayes
YES
Text Data
NO
NOT WORKING
Linear SVC
YES
<100K samples
NO
get more data
NO
>50 samples
YES
regression
SGD Regressor
Lasso
ElasticNet
SVR(kernel='rbf')
EnsembleRegressors
predicting a category
YES
NO
<100K samples
YES
few features should be important
YES
NOT WORKING
RidgeRegression
SVR(kernel='linear')
NO
do you have labeled data
YES
NO
predicting a quantity
YES
NO
clustering
Spectral Clustering
GMM
NOT WORKING
KMeans
YES
number of categories known
YES
NO
<10K samples
NO
MiniBatch KMeans
<10K samples
YES
NO
MeanShift
VBGMM
just looking
YES
NO
Randomized PCA
NOT WORKING
Isomap
Spectral Embedding
NOT WORKING
LLE
YES
<10K samples
NO
kernel approximation
dimensionality reduction
tough luck
predicting structure
Back
scikit learn

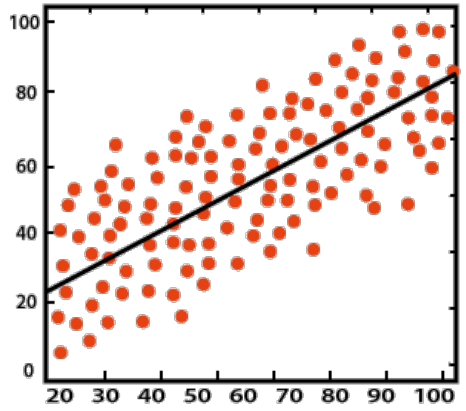
Clasificación vs regresión

- **CLASIFICACIÓN**
  - Variables **discretas** / categóricas
  - Proceso de preparación de los datos, entrenamiento/ajuste y test
  - El objetivo es predecir la **categoría** de nuevos datos



Classification

- **REGRESIÓN**
  - Variables **continuas**
  - Proceso de preparación de los datos, entrenamiento/ajuste y test
  - El objetivo es **modelar** un fenómeno real, o predecir respuesta futura



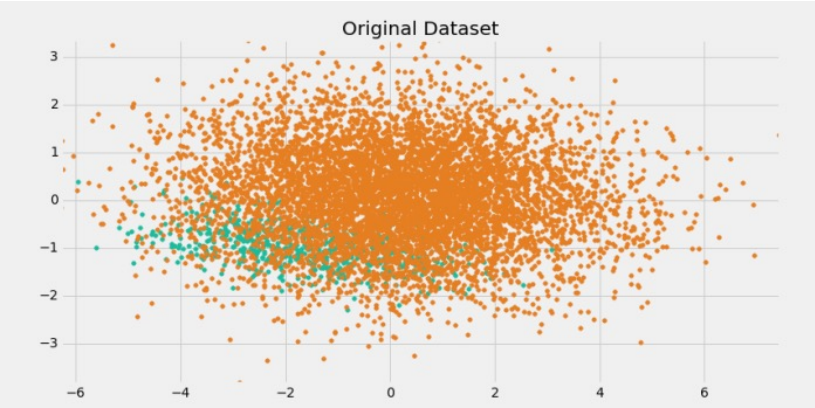
Regression

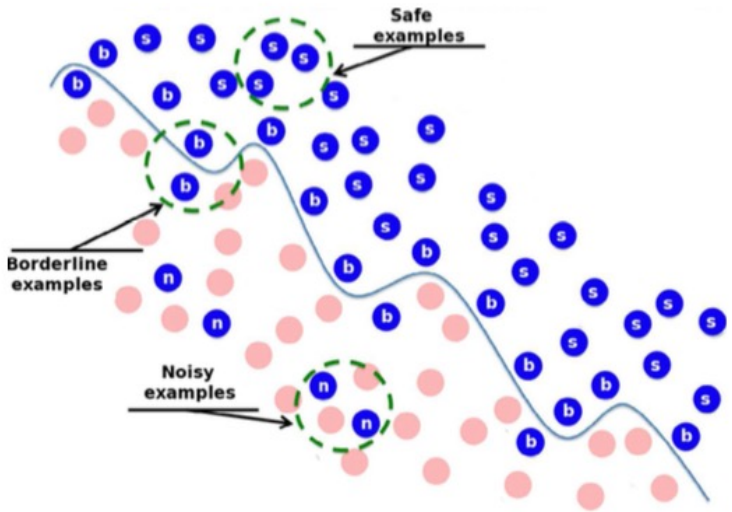
Terminología básica

- Entradas:
  - Vector de entradas: $X = (x_1, x_2, x_3, \ldots x_m) \in \mathcal{R}$
  - *m*: dimensionalidad del espacio de entrada
- Salidas:
  - Vector de predicciones del modelo: $Y = (y_1, y_2, y_3, \ldots y_n) \in \mathcal{R}$
  - *n*: dimensionalidad del espacio de salida
- Objetivos (target):
  - Vector de salidas esperadas: $T = (t_1, t_2, t_3, \ldots t_n) \in \mathcal{R}$
  - *n*: dimensionalidad del espacio de salida
- Error:
  - Función que calcula la diferencia entre la salida del modelo (Y) y los objetivos (T)

Preparación de los datos

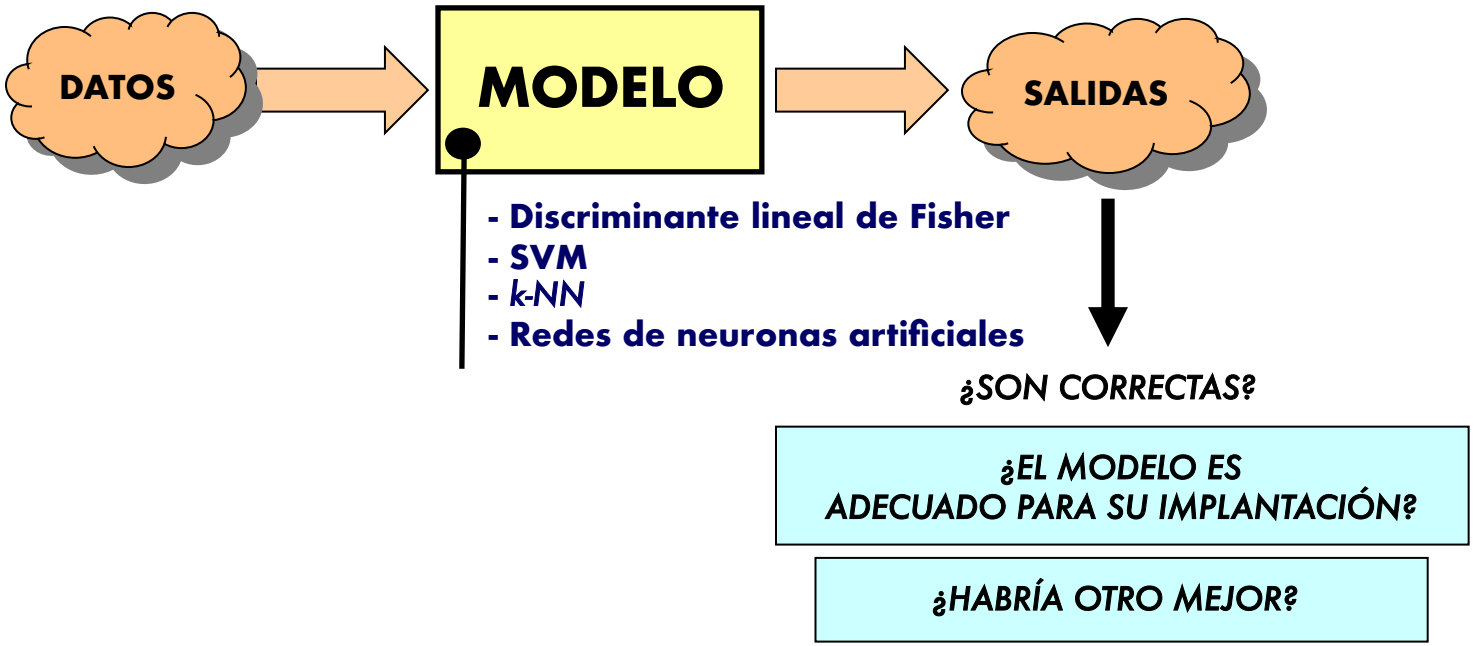
- Antes de aplicar cualquier algoritmo de aprendizaje se deben analizan los datos para:
  - Tratar o eliminar casos con información incompleta
  - Tratar o eliminar, si es posible, casos atípicos y ruido
  - Balancear los conjuntos de datos (número de muestras similares de cada caso)
  - Reescalar los datos para que tengan rangos similares





Análisis de resultados

- Una vez elegido un modelo … ¿cómo podemos estimar su rendimiento?



Análisis de resultados

- El error debería ser estimado sobre toda la población de la que proceden los datos
  - Sin embargo, sólo se dispone de una muestra limitada

- Solución más simple:
  - **Emplear todo el conjunto de datos para entrenar el modelo y para estimar el error**

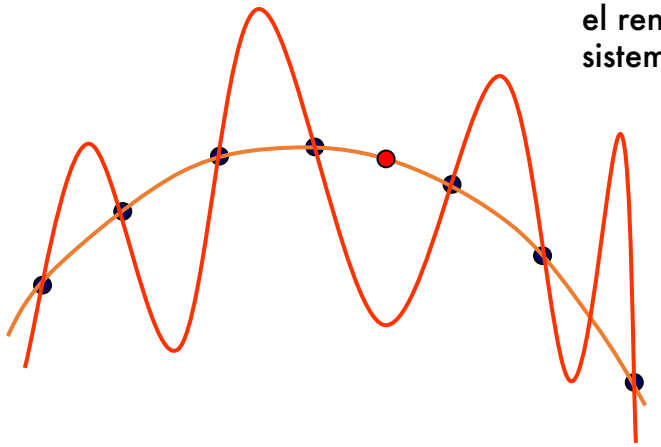
- Problemas:
  - **El modelo obtenido probablemente sobreajustará los datos**
  - **El error obtenido será muy optimista**

Error empírico (entrenamiento) = 0 ⇨ Error ***optimista***

Error real (prueba) > 0

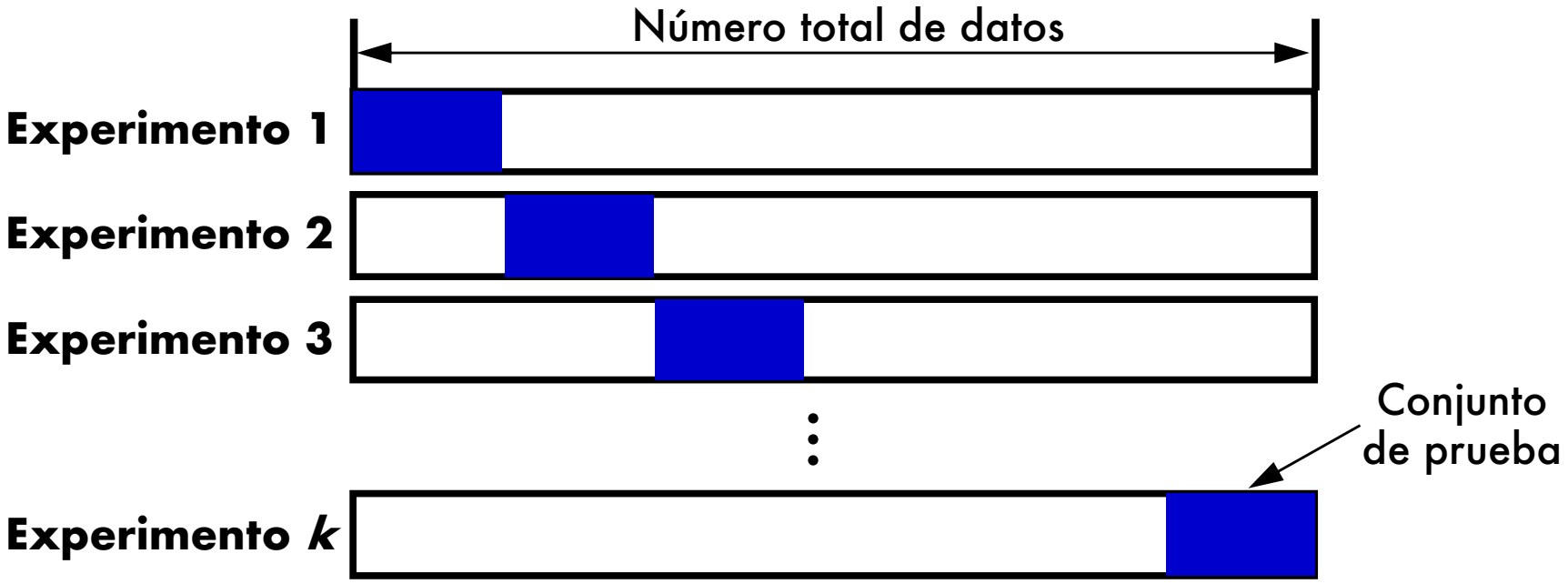
⇩

No es válido para conocer el rendimiento ***real*** del sistema



Validación cruzada *K*-fold

- Soluciones:



Análisis de resultados

a) ¿Qué resultados mostrar? (para regresión):
   - Error medio del cada uno de los conjuntos (entrenamiento y test) ± desviación típica

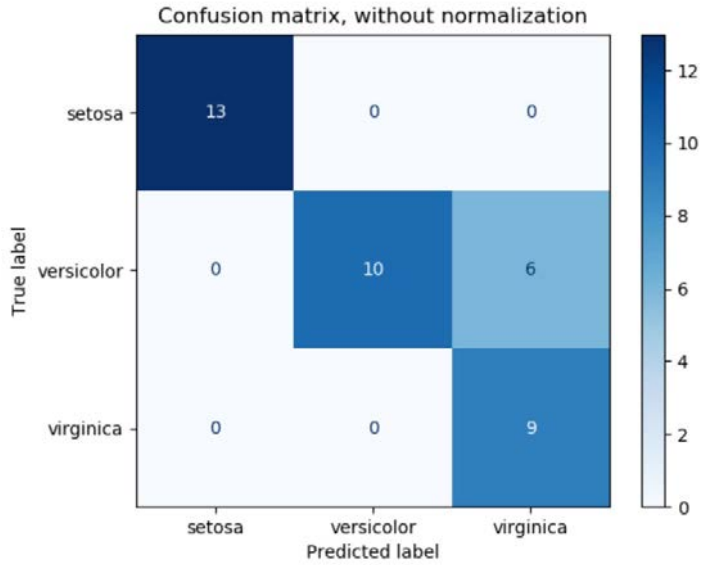
	Error de entrenamiento	**Error de prueba**
Modelo 1	$2{,}86\times10^{-4} \pm 1{,}34\times10^{-6}$	$3{,}87\times10^{-4} \pm 1{,}91\times10^{-6}$
Modelo 2	$2{,}63\times10^{-4} \pm 1{,}17\times10^{-6}$	$4{,}12\times10^{-4} \pm 2{,}09\times10^{-6}$
Modelo 3	$3{,}21\times10^{-4} \pm 2{,}11\times10^{-6}$	$5{,}19\times10^{-4} \pm 3{,}19\times10^{-6}$
Modelo 4	$1{,}93\times10^{-4} \pm 1{,}02\times10^{-6}$	$2{,}98\times10^{-4} \pm 1{,}23\times10^{-6}$

Análisis de resultados

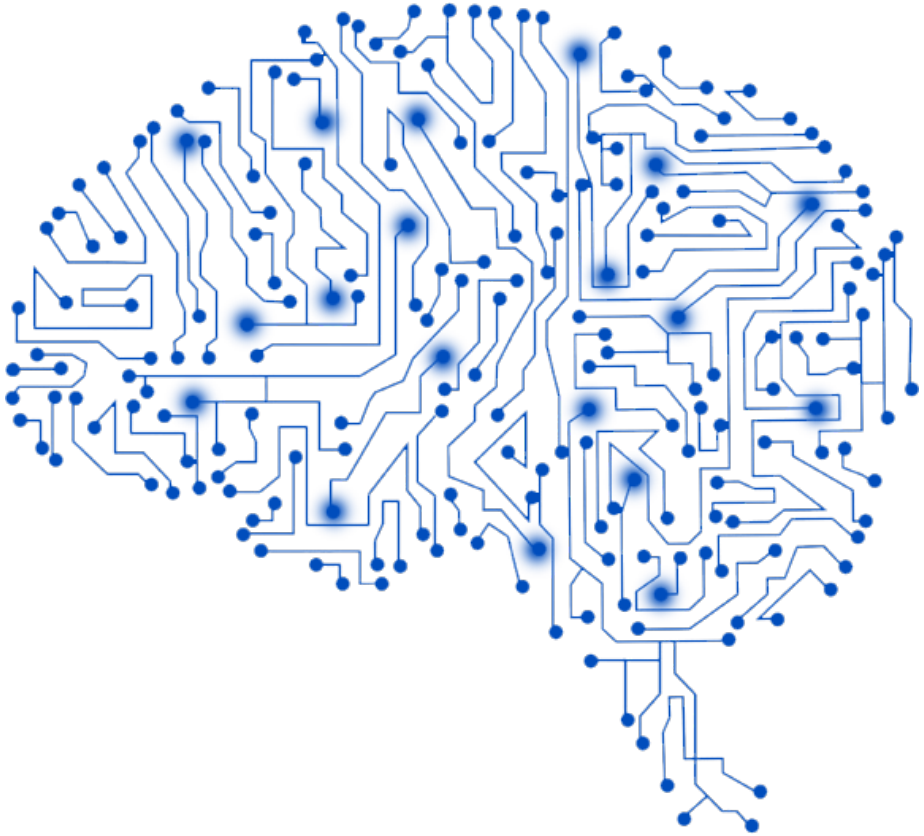
a) ¿Qué resultados mostrar? (para clasificación):

- Matriz de confusión para problemas de dos clases

Resultado del clasificador	Valor real: Positivo	Valor real: Negativo
Positivo	Verdaderos Positivos (VP)	Falsos Positivos (FP)
Negativo	Falsos Negativos (FN)	Verdaderos Negativos (VN)

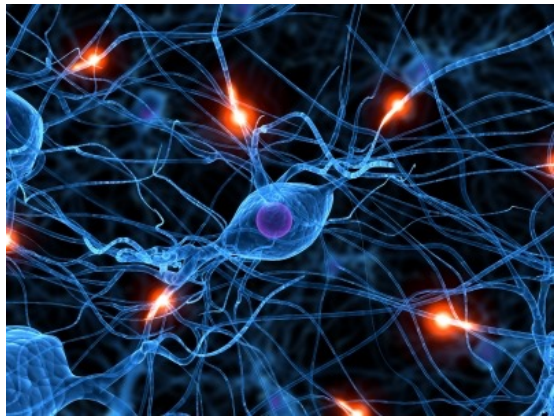


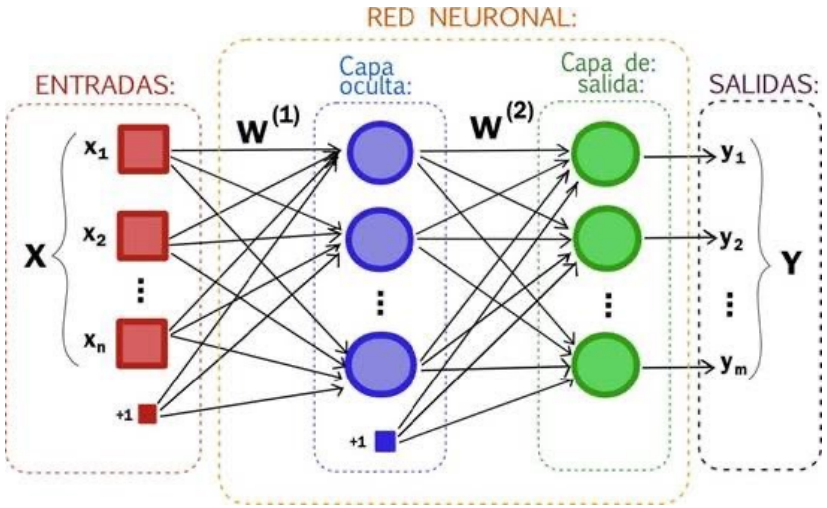
Redes de neuronas artificiales



Introducción

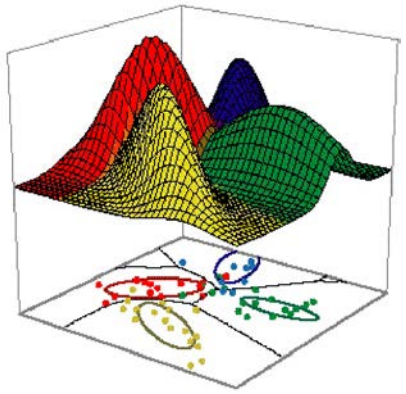
- Las redes de neuronas artificiales (RNA) son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso de los animales.
  - Interconexión de neuronas en una red que produce un estímulo de salida.
  - Generan modelos matemáticos de alta dimensionalidad y capaces de generalizar patrones no lineales

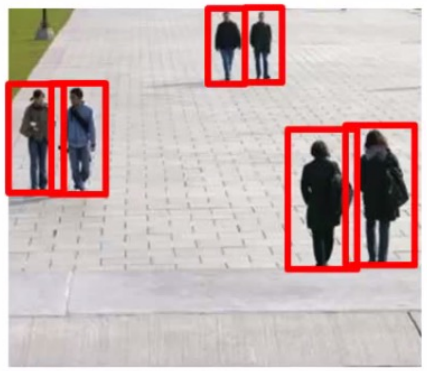


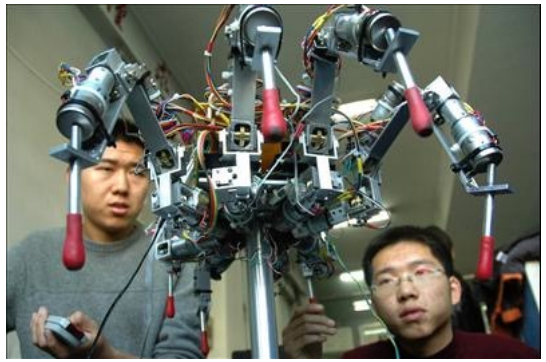


Aplicaciones

- **Clasificación**: reconocimiento de patrones y secuencias, detección de novedades y toma de decisiones
- **Procesado de datos**: filtrado, clusterización, separación de señales y compresión
- **Aproximación de funciones o análisis de regresión**: series temporales, modelado, etc.
- **Robótica**: control numérico, dirección de manipuladores, etc

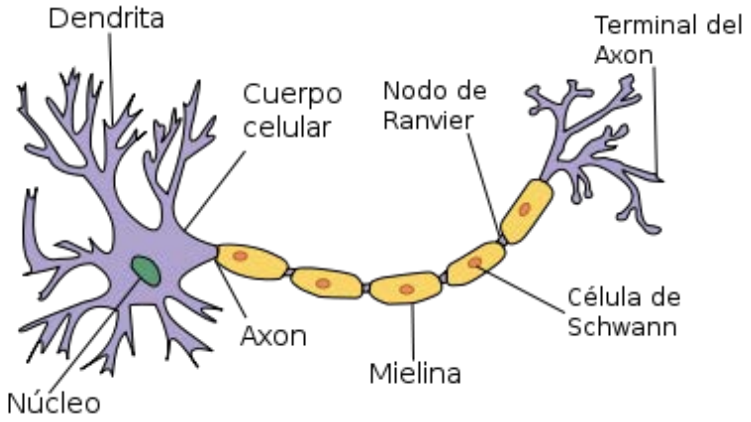




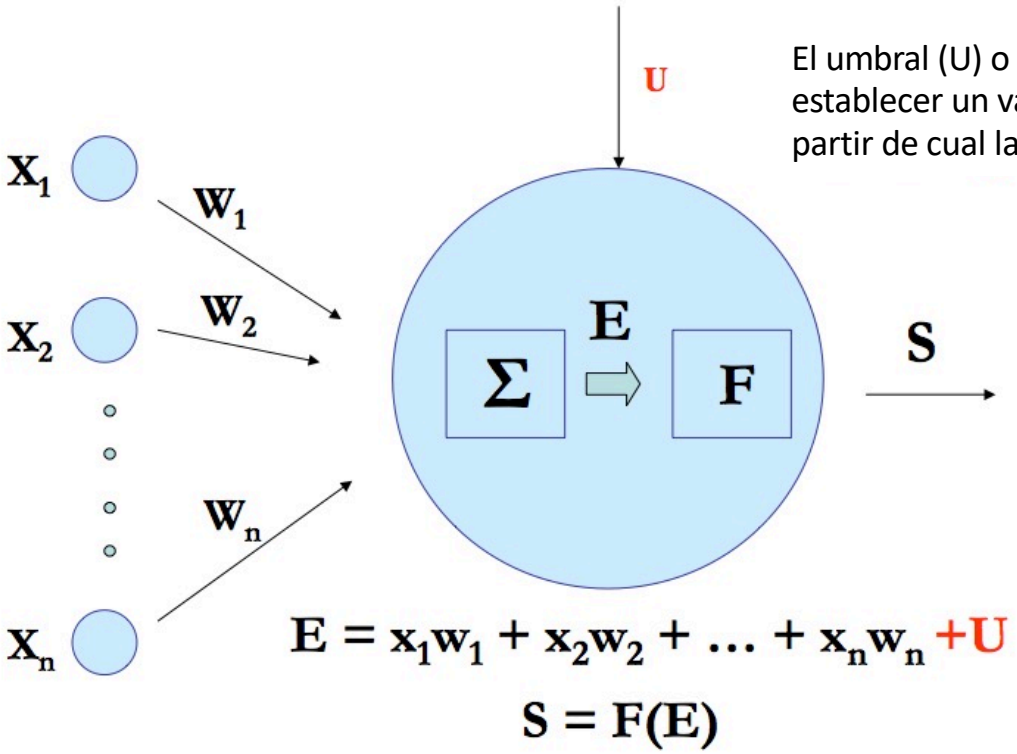


Modelo biológico: la neurona

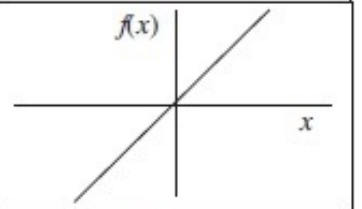
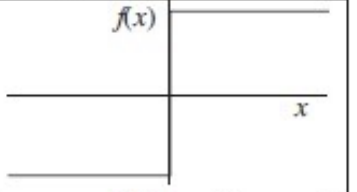
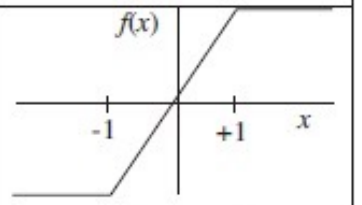
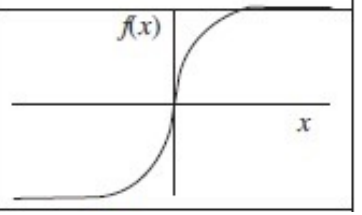
- La neurona es la unidad fundamental del sistema nervioso y en particular del cerebro.
- Cada neurona es una simple unidad procesadora que recibe y combina señales desde y hacia otras neuronas.
- Si la combinación de entradas es suficientemente fuerte, la salida de la neurona se activa.



Modelo computacional

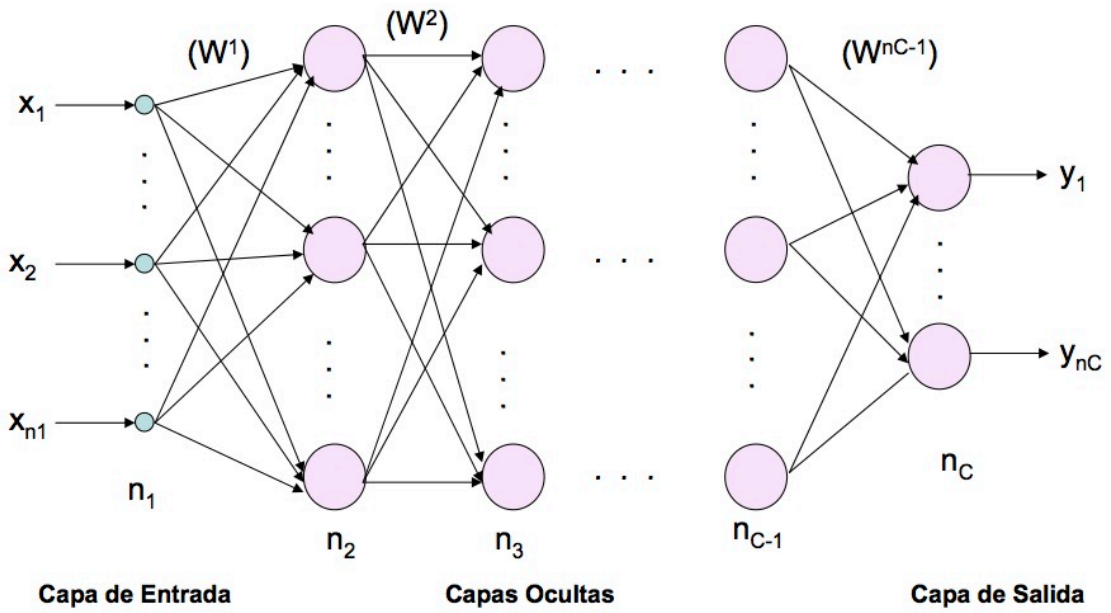


El umbral (U) o bias permite establecer un valor mínimo a partir de cual la red se activa

	Función	Rango	Gráfica
Identidad	$y = x$	$[-\infty,+\infty]$	
Escalón	$y = sign(x)$ $y = H(x)$	$\{-1,+1\}$ $\{0,+1\}$	
Lineal a tramos	$y = \begin{cases} -1, si\ x < -l \\ x, si\ +l \le x \le -l \\ +1, si\ x > +l \end{cases}$	$[-1,+1]$	
Sigmoidea	$y = \frac{1}{1+e^{-x}}$ $y = tgh(x)$	$[0,+1]$ $[-1,+1]$	

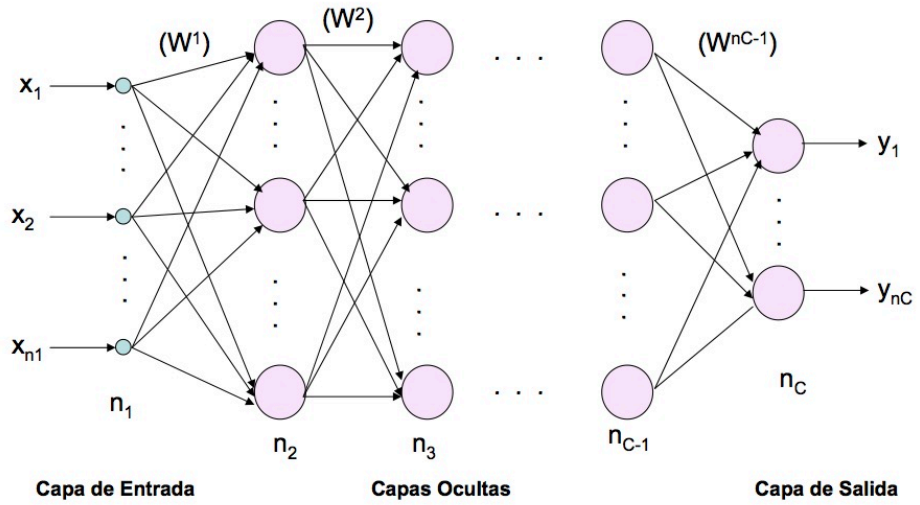
Estructura básica de la red

- Arquitectura: forma de conexión entre neuronas
- La estructura básica: red multicapa
  - Capa de entrada
  - Capas ocultas
  - Capa de salida
- Pesos (W_{ij})

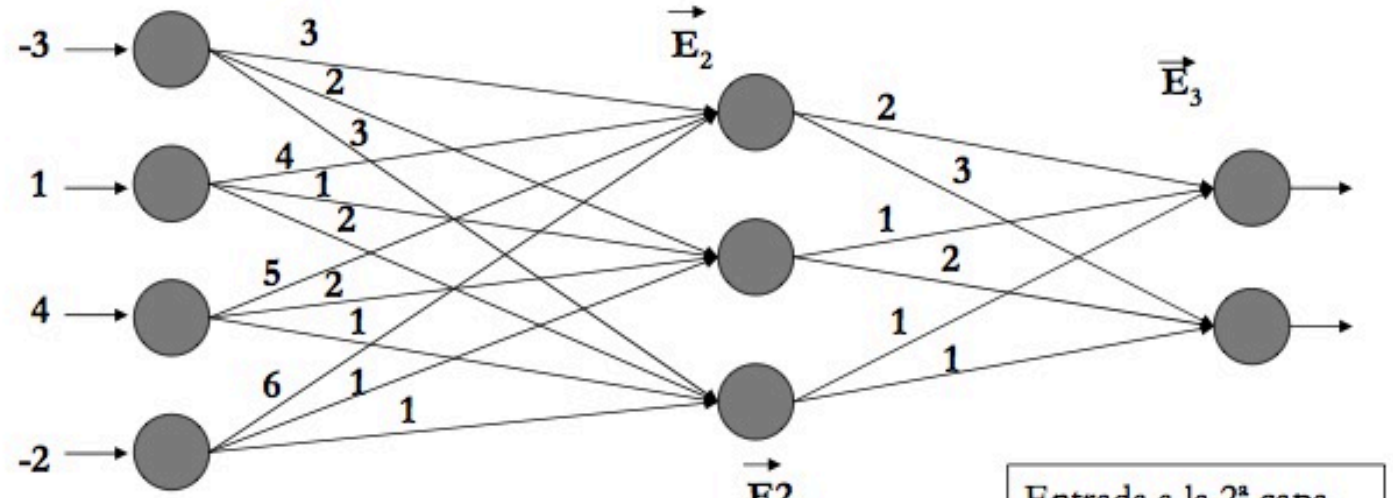


Estructura básica de la red

- Propagación de la señal hacia delante
  - Existen redes con propagación hacia atrás
- El valor de la señal se modifica según el peso de la conexión
- Los pesos se ajustan en la fase de aprendizaje



Perceptrón multicapa. Ejemplo de ejecución



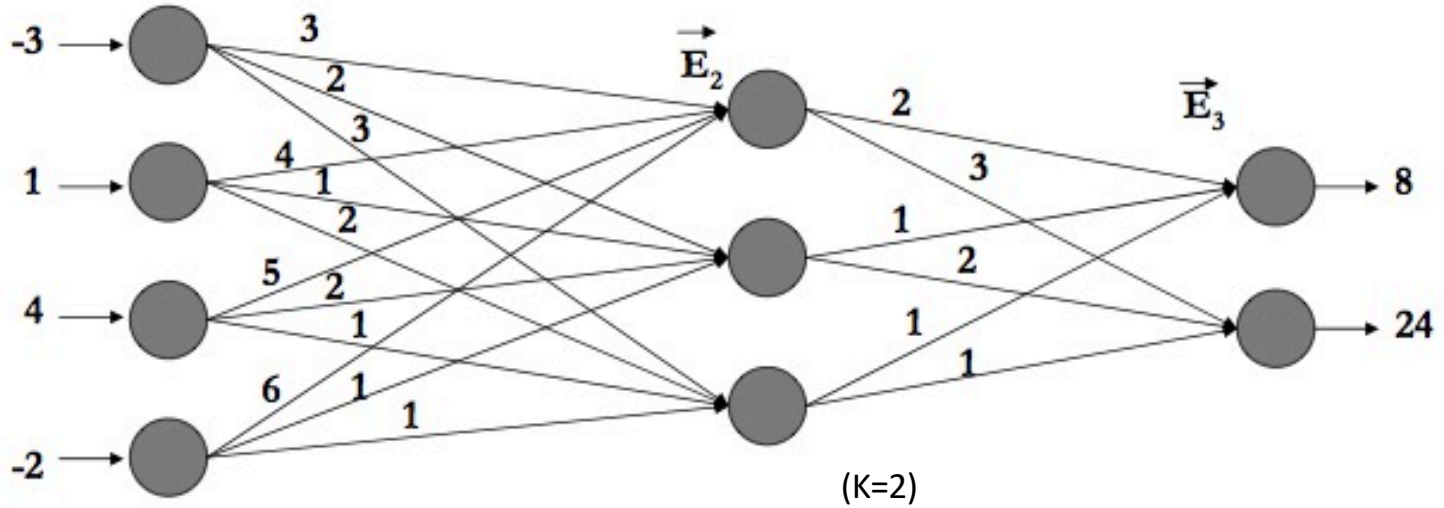
$\vec{E2}$

$$\vec{S} = F(F(\vec{x} \bullet w_1) \bullet w_2)$$

Entrada a la 2ª capa

$$\vec{E}_2 = \vec{x} \bullet w_1 = \begin{pmatrix} -3 & 1 & 4 & -2 \end{pmatrix} \begin{pmatrix} 3 & 2 & 3 \\ 4 & 1 & 2 \\ 5 & 2 & 1 \\ 6 & 1 & 1 \end{pmatrix} = \vec{E2} \begin{pmatrix} 3 & 1 & -5 \end{pmatrix}$$

Perceptrón multicapa. Ejemplo de ejecución



(K=2)

$$F(x) = k \bullet x$$

$$\vec{S} = F(F(\vec{E}_2) \bullet w_2) = F(\begin{bmatrix} 6 & 2 & -10 \end{bmatrix} \begin{bmatrix} 2 & 3 \\ 1 & 2 \\ 1 & 1 \end{bmatrix}) = F(\begin{bmatrix} 4 & 12 \end{bmatrix}) = \begin{bmatrix} 8 & 24 \end{bmatrix}$$

$\vec{E}_3$ (sobre $\vec{E}_2$) $\vec{E}_3$

Aprendizaje

- **Ajuste de los parámetros (pesos, umbral, etc) para la minimización de una función de error**

- El proceso de aprendizaje consiste en:
  1. Introducir paulatinamente todos los ejemplos de aprendizaje
  2. Modificar los pesos siguiendo un esquema de aprendizaje
  3. Cuando se han introducido todos, se comprueba si se cumple cierto criterio de convergencia. Si no se cumple, se repite el proceso.

Backpropagation

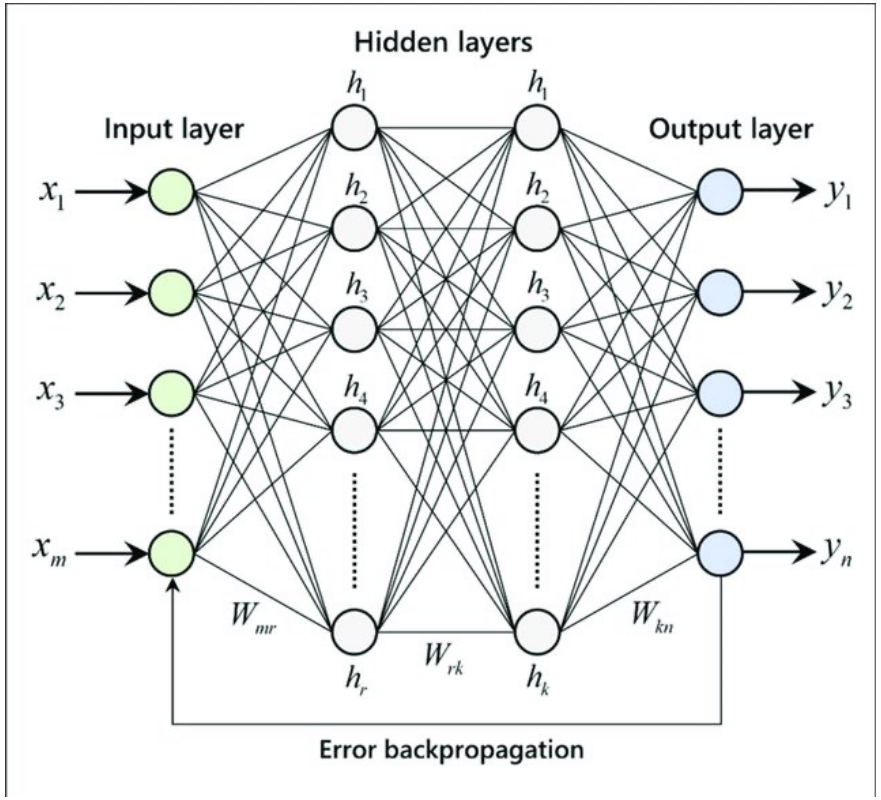
- Dado un vector ***x*** de entradas
- Una matriz de pesos de la primera capa ***v***
- Un vector ***y*** de salidas de la red
- Un vector ***t*** de salidas deseadas (target)
- Una función de activación sigmoide ***g***:

$$a = g(h) = \frac{1}{1 + \exp(-\beta h)}$$

- Una función de error ***E***:

$$E(\mathbf{t}, \mathbf{y}) = \frac{1}{2} \sum_{k=1}^{N} (y_k - t_k)^2$$

https://www.youtube.com/watch?v=odlgtjXduVg



SGD y Adam

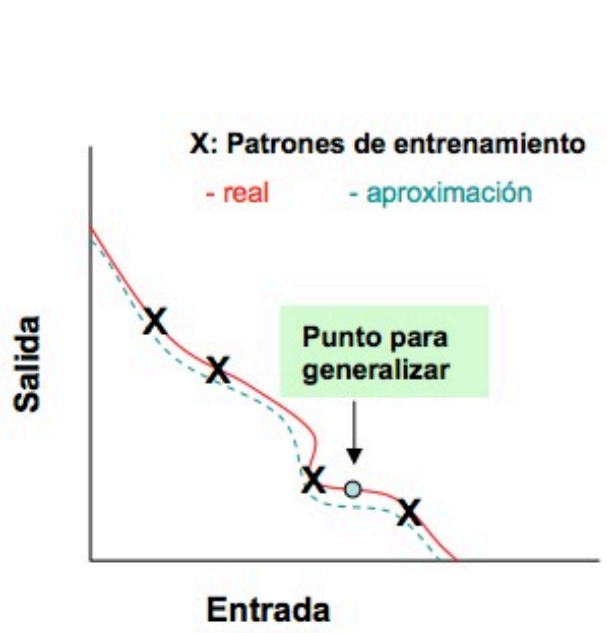
- Son los 2 algoritmos de entrenamiento más habituales:
  - **SGD** utiliza la misma tasa de aprendizaje durante el entrenamiento.
  - **Adam** utiliza una tasa de aprendizaje para cada peso (parámetro) de la red, que se adapta por separado a medida que se desarrolla el aprendizaje.

MLP: criterios de parada

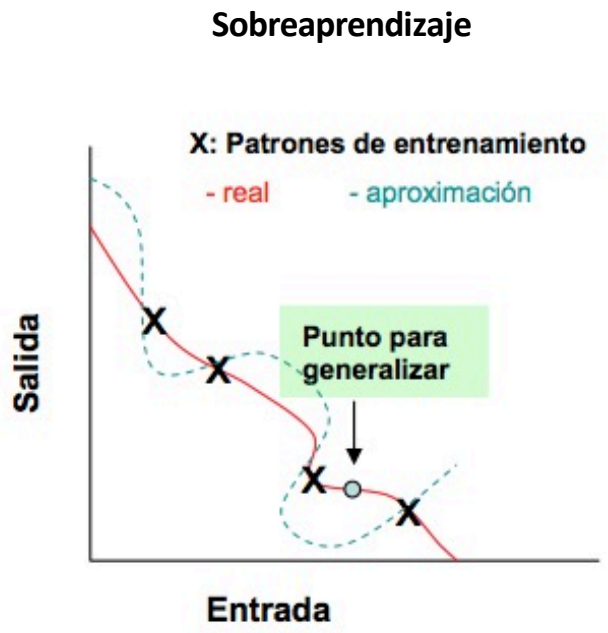
- Número de pasos de entrenamiento
  - Prueba y error
- El error promedio está por debajo de un umbral
  - Umbral propio de cada problema
  - Difícil de estimar
- El error se estabiliza
  - Criterio más común
  - Para el entrenamiento cuando no hay mejora

MLP: generalización

- Como en cualquier método supervisado
  - Capacidad de generalización se evalúa con un conjunto de datos de test



Sobreaprendizaje



MLP: generalización

- El sobreaprendizaje en el MLP puede deberse a:
  - Un número excesivo de pasos de aprendizaje
  - Un excesivo número de pesos o neuronas ocultas
  - Complejidad del modelo alta en relación con el número de patrones de entrenamiento
  - Patrones mal balanceados y normalizados

MLP: diseño

- Se desordenan los datos
- Se normalizan los datos
- El conjunto de datos disponibles se divide en 2 subconjuntos:
  - Entrenamiento (70%)
    - Usado para ajustar el valor de los pesos
  - Test (30%)
    - Usado para medir la eficacia de la red

MLP: diseño

- **Selección de la función de activación**
- **Selección del número de neuronas de entrada y de salida**
  - Vienen dados por las variables que definen el problema
- **Selección del número de capas y neuronas ocultas**
  - Debe ser elegido por el diseñador
- **Selección del algoritmo de aprendizaje**
  - Parametrizar el algoritmo
- Utilizar **validación cruzada** y un **análisis formal** de los datos
  - Medidas de error

Bibliografía

- La guía de recursos adaptados para el Bloque II – Parte II proporcionada como material adjunto a esta sesión contiene una relación exhaustiva de bibliografía adaptada

- Recursos prácticos recomendados:
  - AI and Machine Learning en code.org (enlace)
  - Teachable Machine (enlace)
  - Machine Learning con Scratch (enlace)

Recursos prácticos

Indice

Quickdraw

- https://quickdraw.withgoogle.com
- Red neuronal que reconoce dibujos
- Interesante con estudiantes jóvenes o como muestra de concepto
- Utiliza nuestros dibujos para continuar el entrenamiento

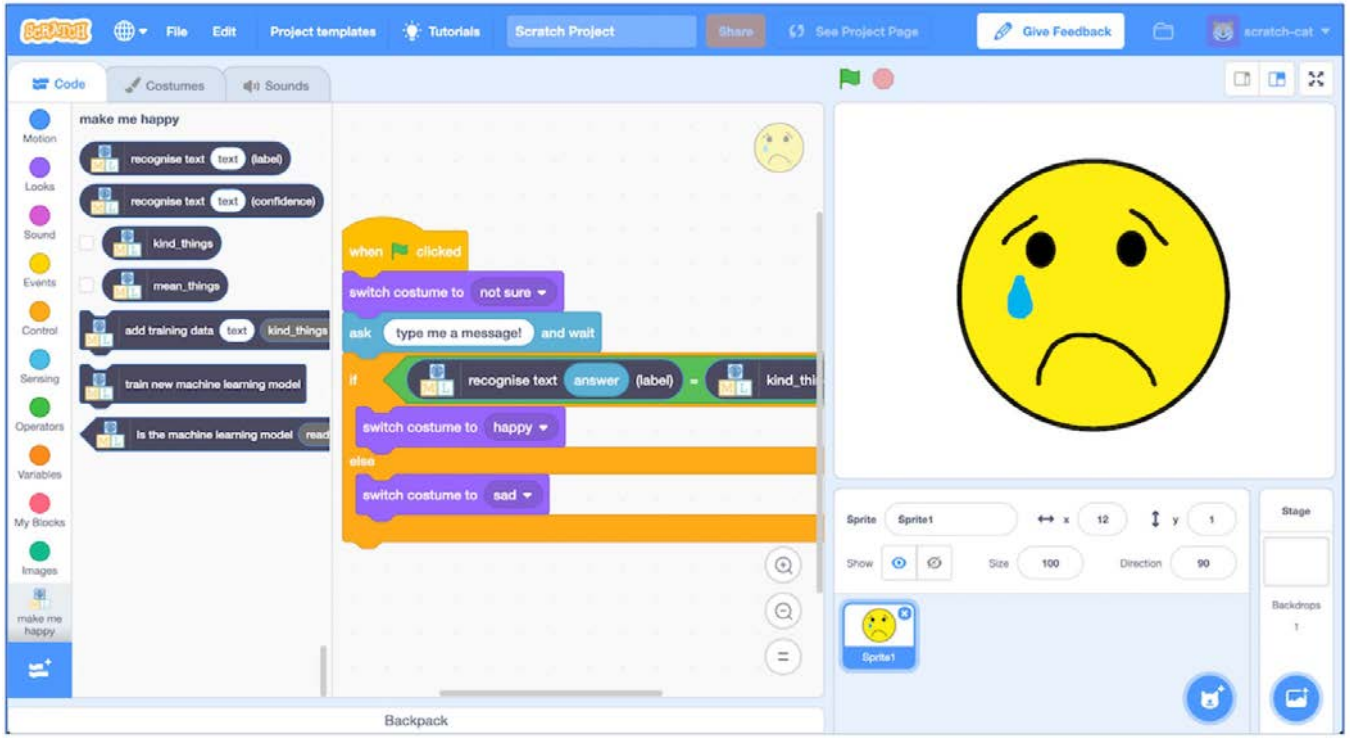
Machine Learning for kids

- https://machinelearningforkids.co.uk/
  - Introducción al aprendizaje automático a través de experiencias prácticas para entrenar sistemas de aprendizaje automático y construir cosas con ellos.
  - Entorno de aprendizaje guiado para **entrenar modelos de aprendizaje automático capaces de identificar texto, números o imágenes**.
  - Scratch o Python

ML4K: Make me happy

- **Actividad:** Crea un personaje en Scratch que sonría si le dices cosas buenas y que llore si le dices cosas malas.
- **Objetivo:** Enseñar a un ordenador a reconocer cumplidos e insultos
  - Cómo se pueden entrenar los ordenadores para reconocer el tono emocional
  - Cómo el aprendizaje supervisado construye sistemas que pueden lidiar con entradas inesperadas
- **Resumen:** Los estudiantes entrenarán un modelo de aprendizaje automático para reconocer los cumplidos y los insultos escribiendo ejemplos de frases amables y desagradables. Lo utilizarán en Scratch para crear un personaje que reaccione a los mensajes en función del sentimiento.

ML4K: Make me happy



Aprendizaje con App Inventor

Co-funded by the
Erasmus+ Programme
of the European Union

Crear un juego de búsqueda de objetos

- Crearemos una red de neuronas que reconozca 3 objetos


Preparación de datos → Entrenamiento del modelo → Testeo del modelo → Uso del modelo

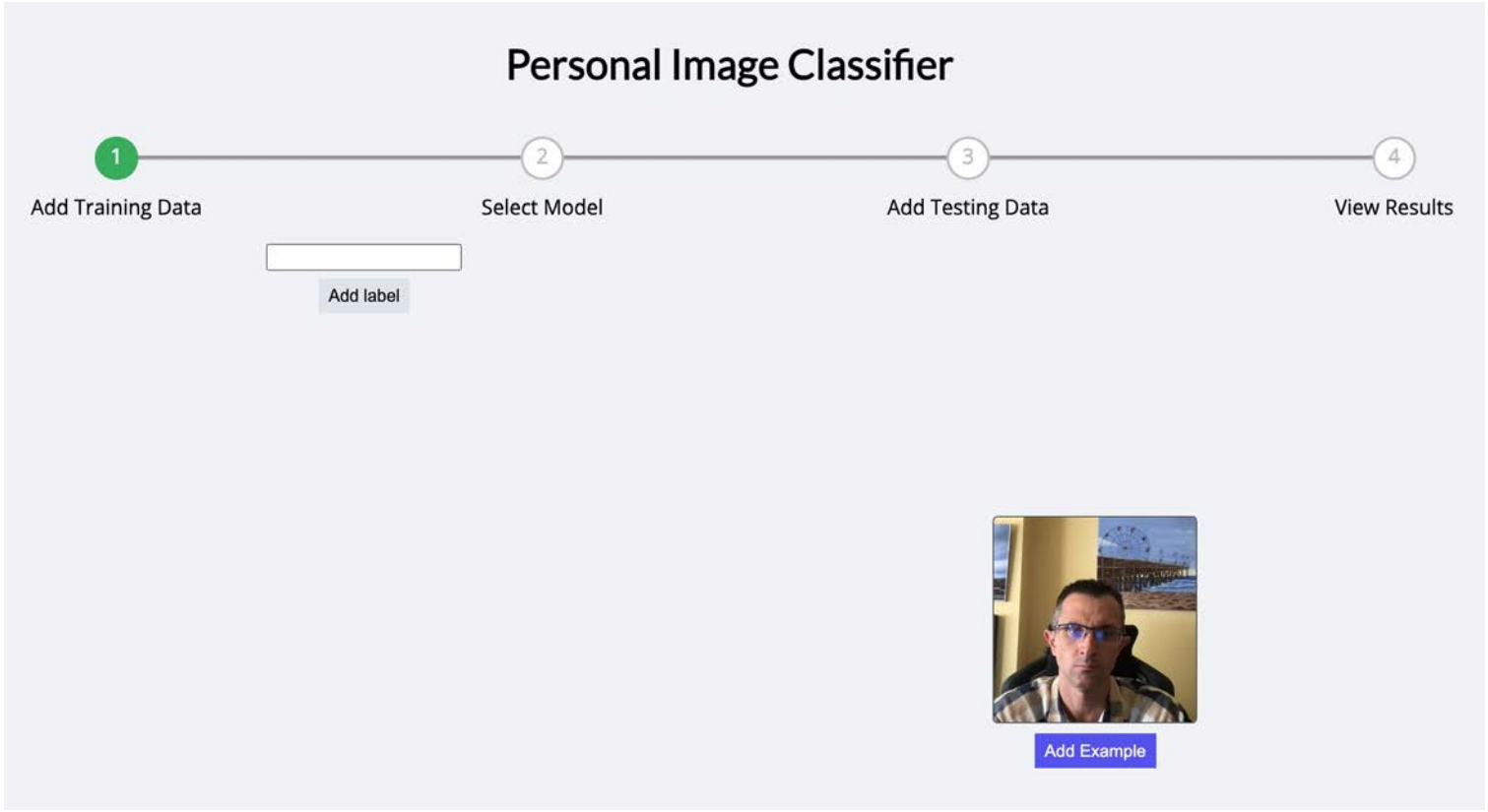
https://www.youtube.com/watch?v=IoMX_qlGFUM

Paso 1: obtener datos

1. Seleccionar los 3 objetos a clasificar
2. Obtener 10 fotos de cada objeto
   - Sacarlas con el smartphone
   - Descargarlas de internet
   - Usar las nuestras:
     - https://drive.google.com/drive/folders/1b0pc2FjbgOcEXbNMdx9sEIc-AAmNo6-q
3. Características de las imágenes:
   - Sin fondo
   - Objeto no solapado con otros
   - Diferentes ángulos de visión
4. Guardarlas en el ordenador y poner nombres adecuados

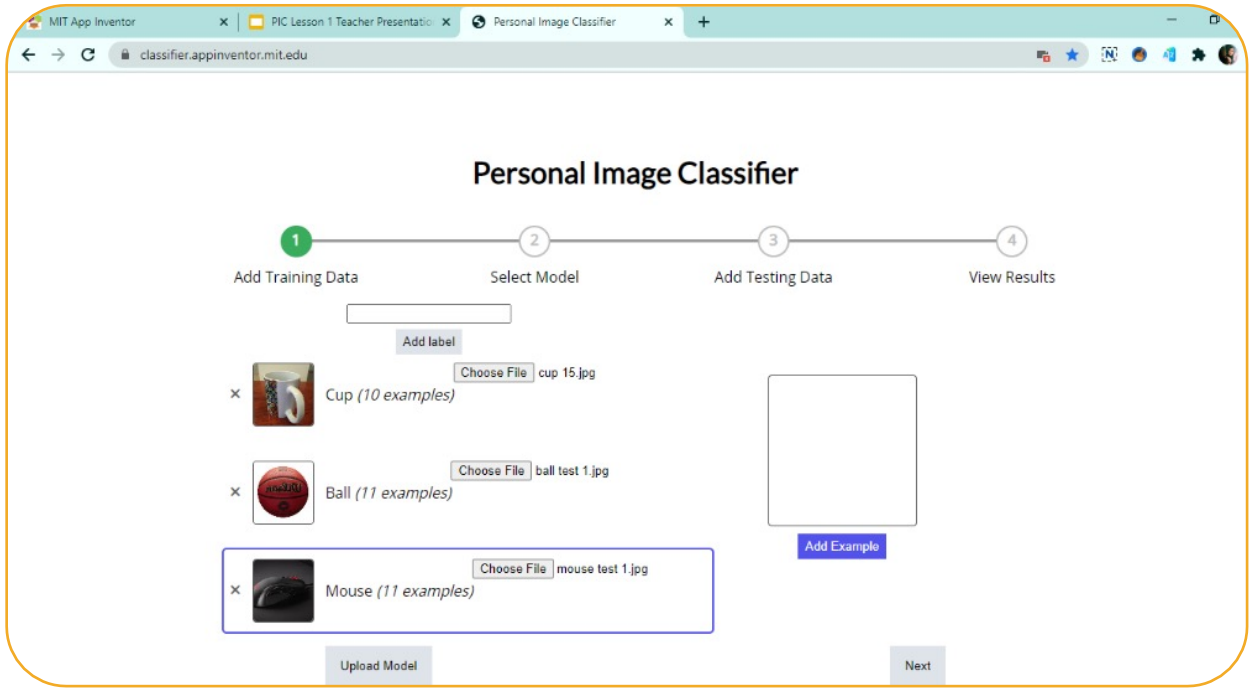
Paso 2: Ir a la página de la extensión

- https://classifier.appinventor.mit.edu/oldpic/

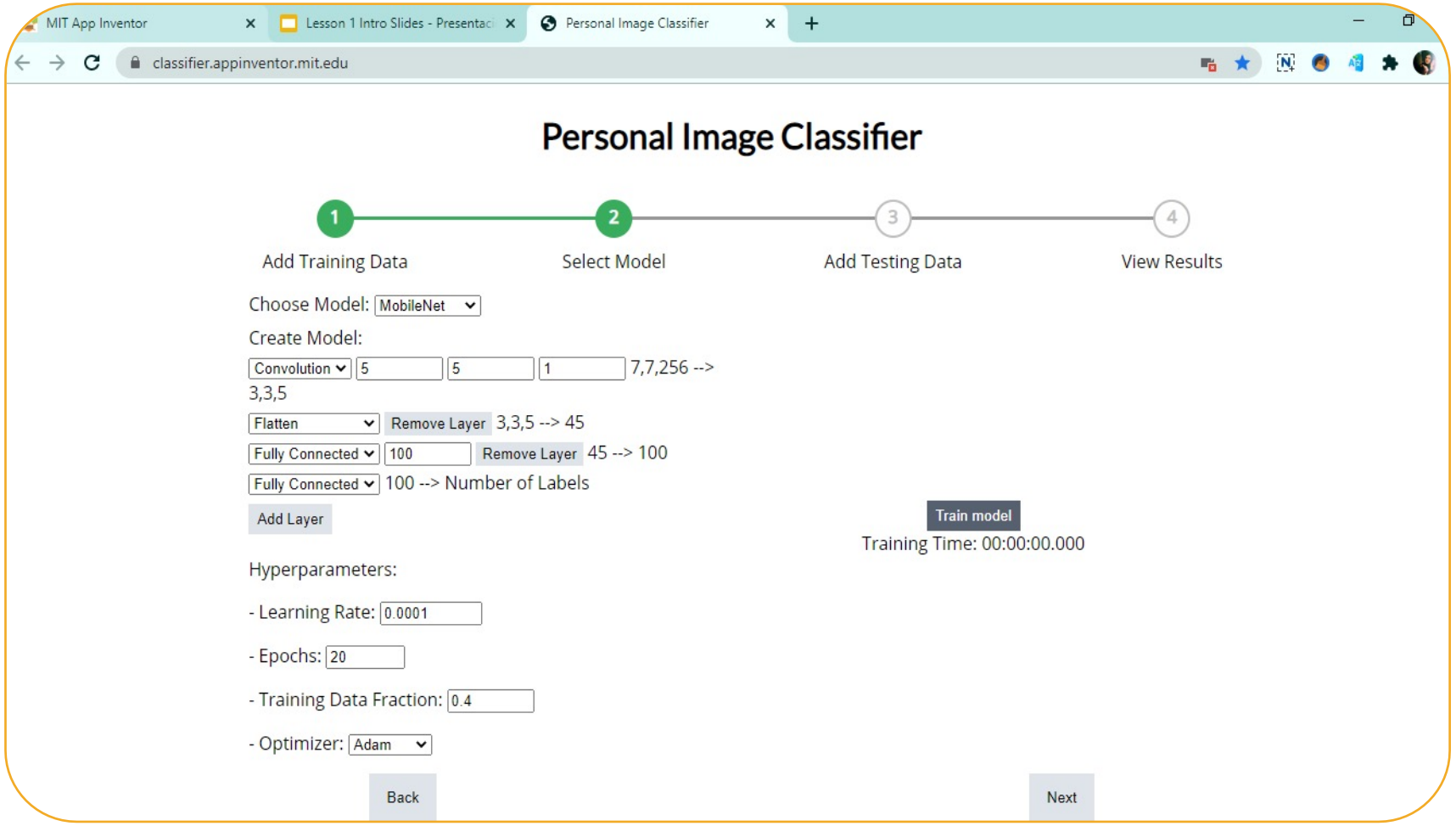


Paso 3: añadir datos de entrenamiento

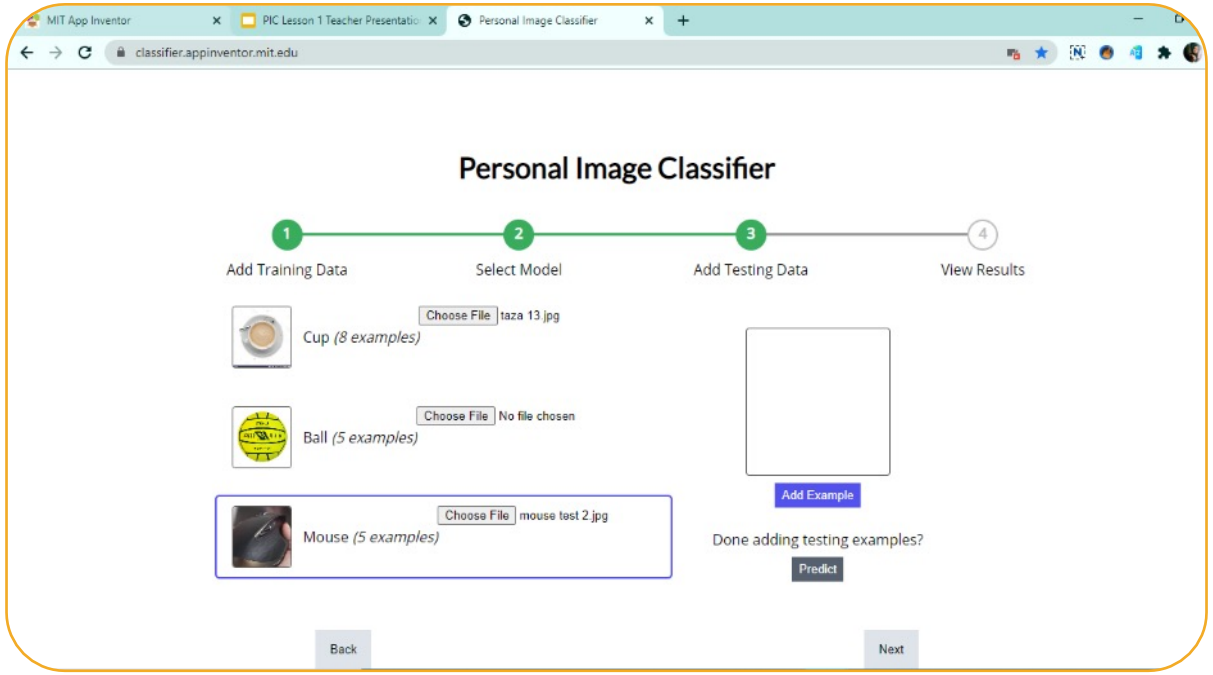
- Los datos de entrada son imágenes de los objetos seleccionados y la salida son etiquetas de texto con el nombre del objeto

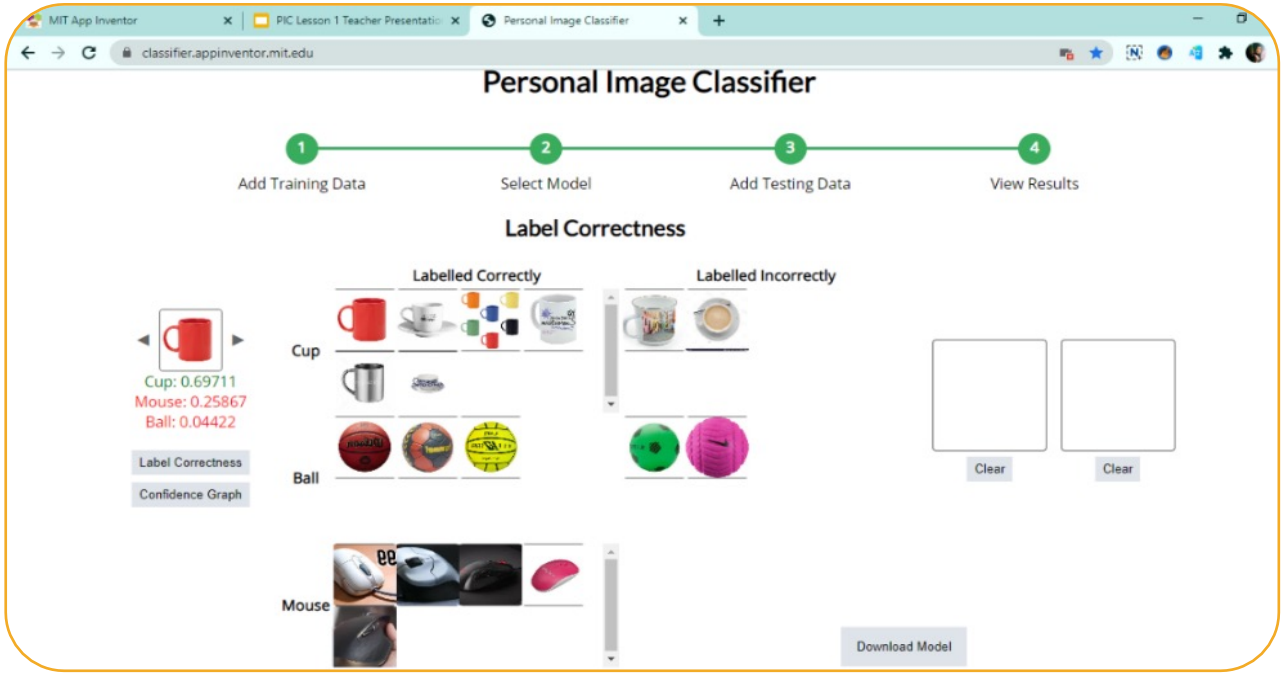


Paso 4: entrenamiento del modelo



Paso 5: añadir datos de test

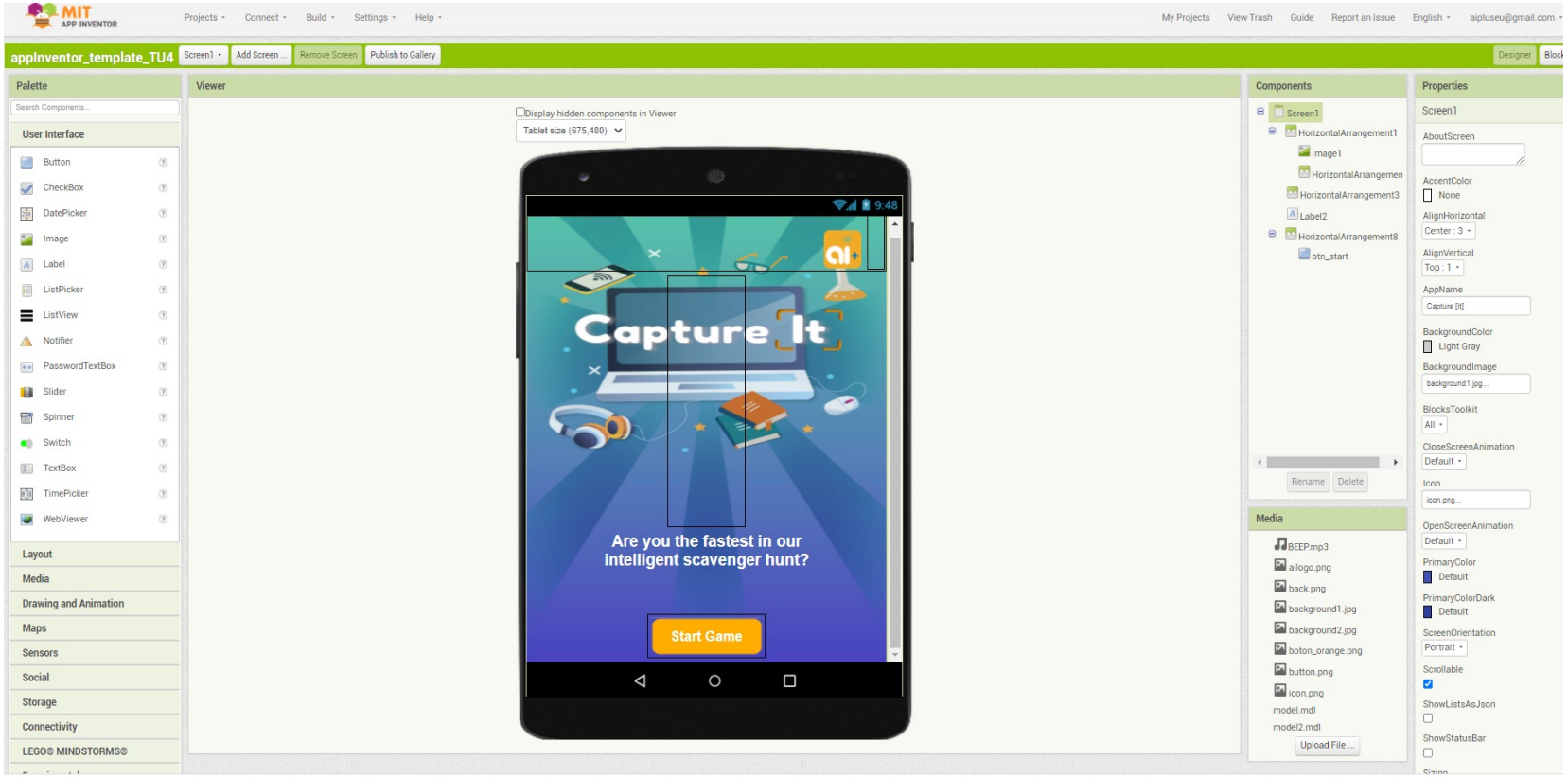




- Si falla muchos casos, debemos probar:
  - Modelo más complejo (capas y neuronas) o más pasos de entrenamiento
  - Mejorar los datos: añadir más casos, añadir mejores casos

Paso 5: utilizar el modelo

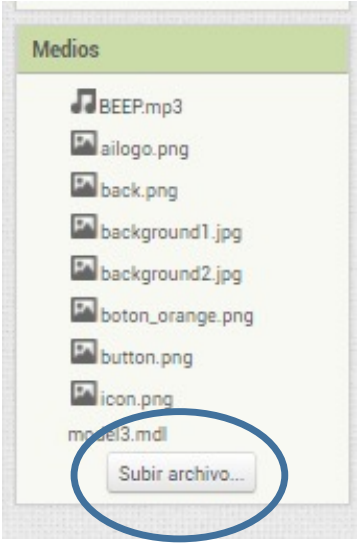
- Abrimos la plantilla subida al drive
  - https://drive.google.com/drive/folders/1b0pc2FjbgOcEXbNMdx9sEIc-AAmNo6-q



Paso 6: utilizar el modelo

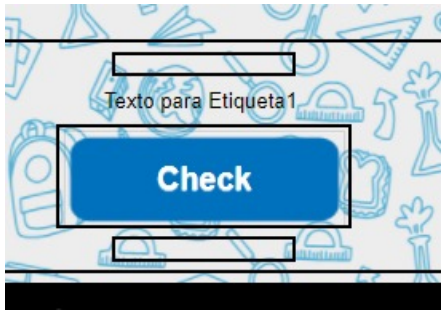
- Paso 1
  - Para añadir el modelo es necesario añadir la extensión PersonalImageClassifier al proyecto de AppInventor.
  - En la plantilla que se os proporciona ya está añadida, pero si se parte de cero está disponible aquí: https://mit-cml.github.io/extensions/

- Paso 2
  - Cargamos el modelo en el menú medios



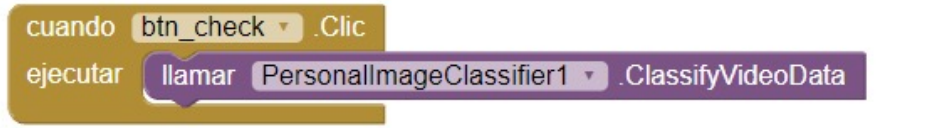
Paso 6: utilizar el modelo

- Paso 3
  - En las propiedades del componente no visible PersonalImageClassifier seleccionamos el modelo
- Paso 4
  - Añadimos una etiqueta para cargar en ella los resultados obtenidos



Paso 6: utilizar el modelo

- Paso 5
  - Programamos botón *check* para que muestre el resultado.



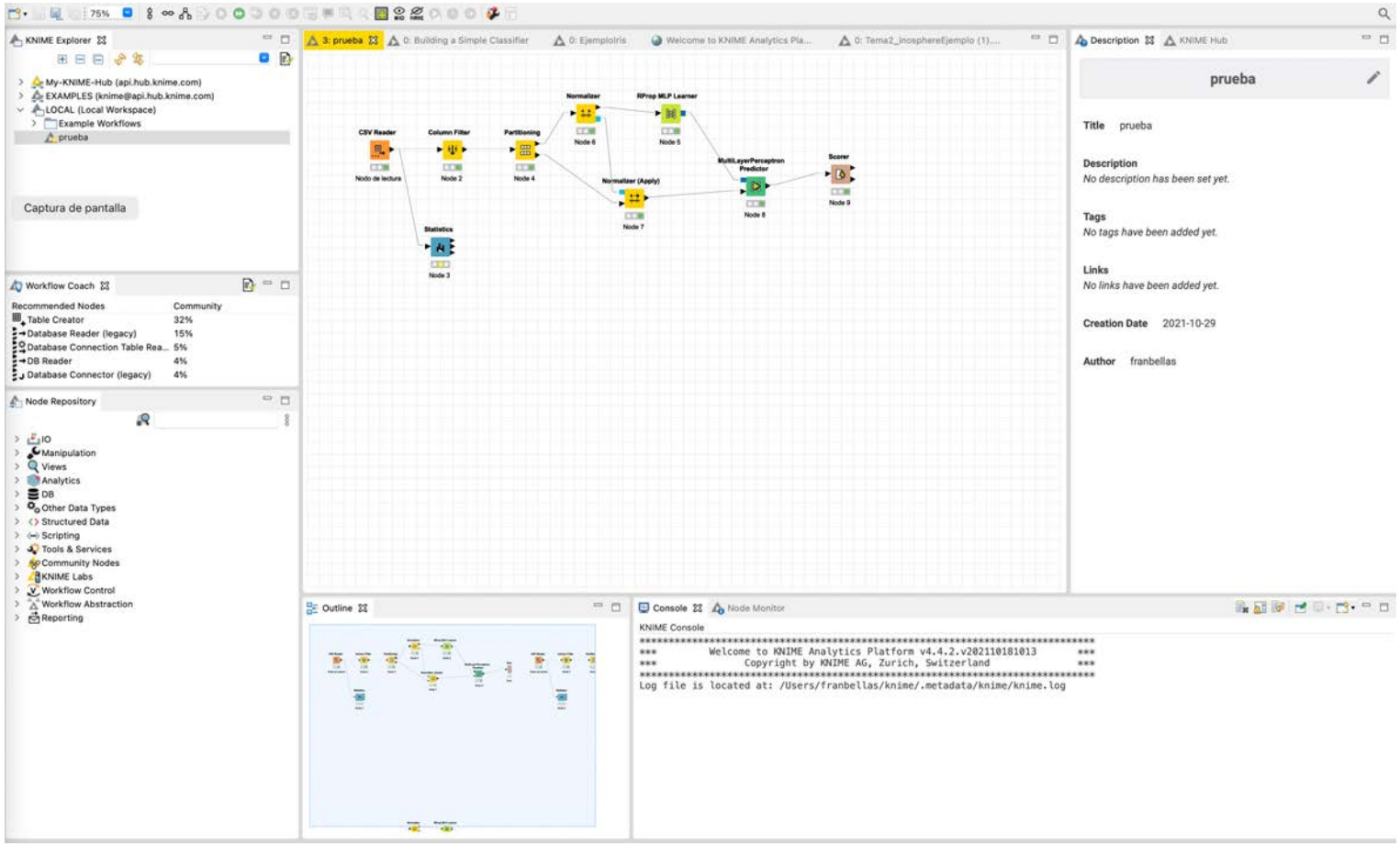
- Paso 6
  - Sería crear un juego similar al de la TU4 del AI+
  - En este enlace encontrarás la actividad mucho más detallada: https://drive.google.com/file/d/1_Hh4Iwk9rIGHe87SwrVuOT5XUoOz9nKv/view?usp=sharing

Aprendizaje con KNIME

- Knime es una herramienta software para análisis de datos
- Permite programar flujos de datos sin conocimientos explícitos de programación
- Descarga gratuita tras registro:
  - https://www.knime.com/downloads

Knime: pasos

1. Arrancar el software
2. Fijar ruta trabajo
3. Localizar:
    1. Explorador
    2. Repositorio de nodos
    3. Área de flujo
    4. Descripción
    5. Consola

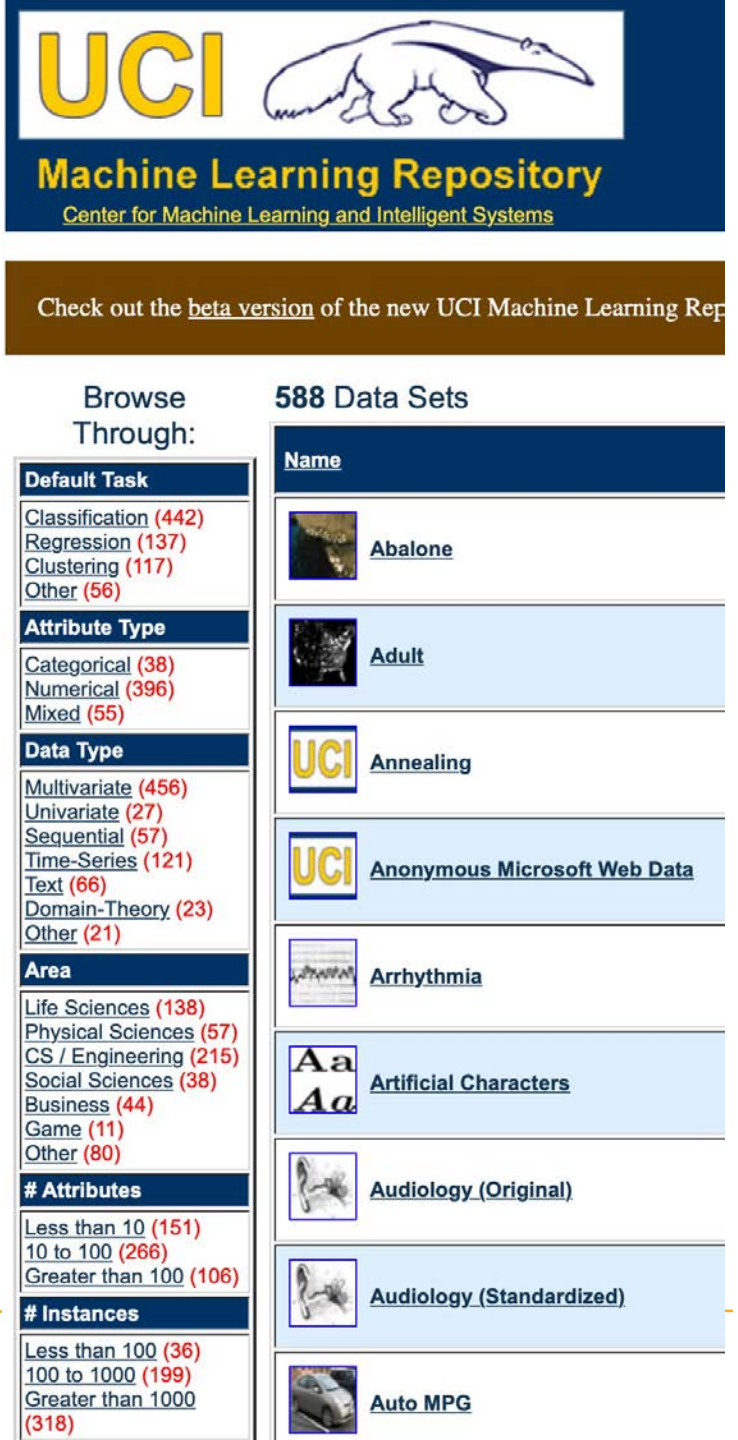


Conjuntos de datos

- Repositorios públicos con datos:
  - https://archive.ics.uci.edu/ml/index.php

- Clasificación
- Regresión
- Agrupamiento

- Organizados por otras categorías



Knime: ionosphere

- Datos de radar recolectados por un sistema en Goose Bay
- Consiste en una serie de 16 antenas de alta frecuencia
- Los retornos de radar se utilizan para estudiar la física de la ionosfera en las capas E y F (100 a 500 km de altitud)
- Los objetivos son electrones libres en la ionosfera.
  - Los "buenos" retornos de radar son aquellos que muestran evidencia de algún tipo de estructura en la ionosfera.
  - Las devoluciones "malas" son aquellas que no lo hacen

Material del curso:
https://drive.google.com/drive/folders/1ck6eBmDMR-CsH9iISx7QZwcZCxsUEcmD

Knime: ionosphere

- Los datos se describen mediante 2 atributos por cada número de pulso, correspondientes a los valores reales y complejos devueltos por la función que resulta de la señal electromagnética
  - https://archive.ics.uci.edu/ml/datasets/ionosphere

- **Variables de entrada**: 34

- **Variable a clasificar**: g ó b (good/bad)

- **Numero de datos**: 351

Knime: ionosphere

1. Abrir archivo .csv y ver su contenido

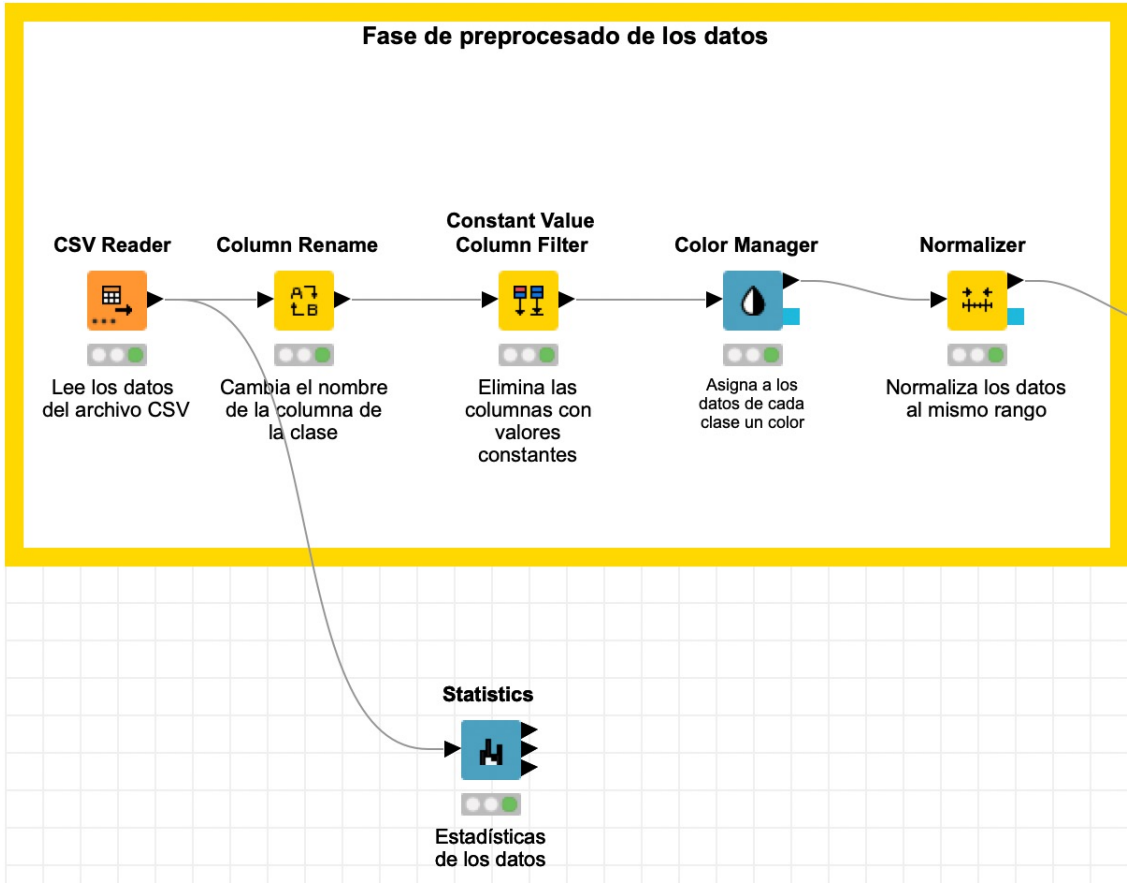
2. Comprender la estructura y los rangos de los datos
   - ¿Tienen cabecera?

3. Comprender el problema de clasificación a resolver

4. Establecer unos rangos de validez del resultado

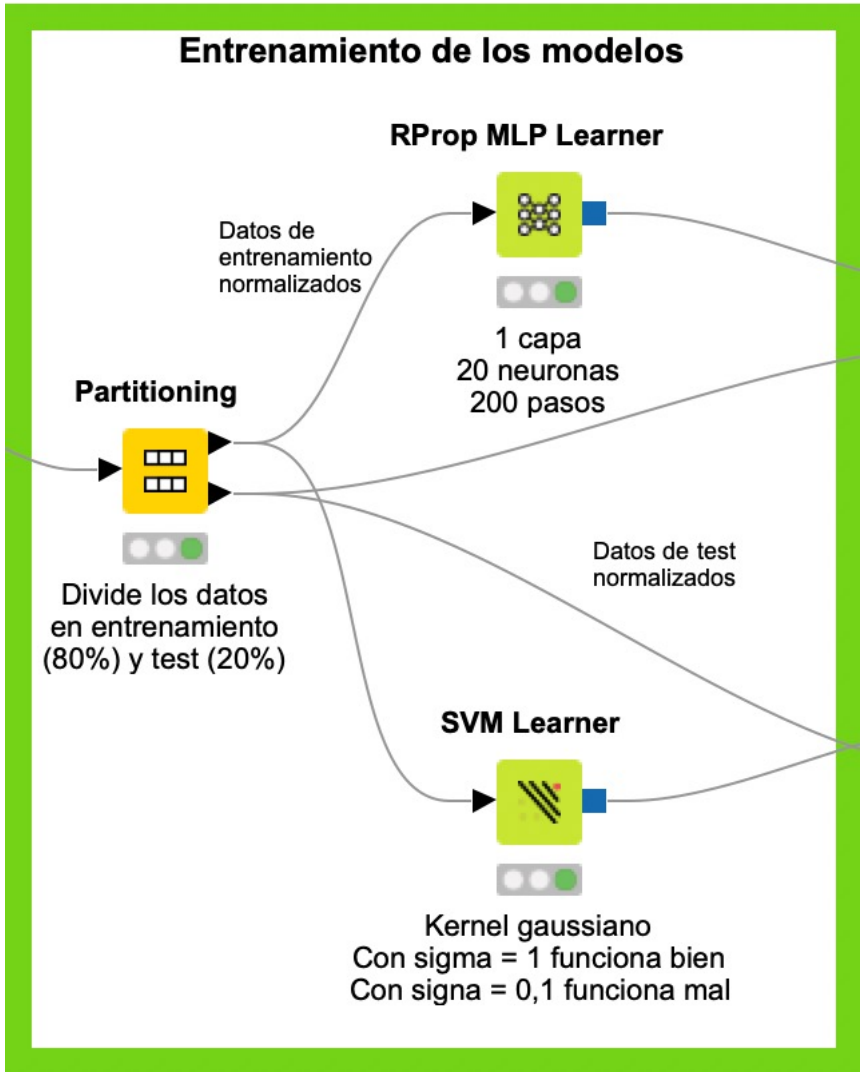
Knime: ionosphere

1. Preprocesado



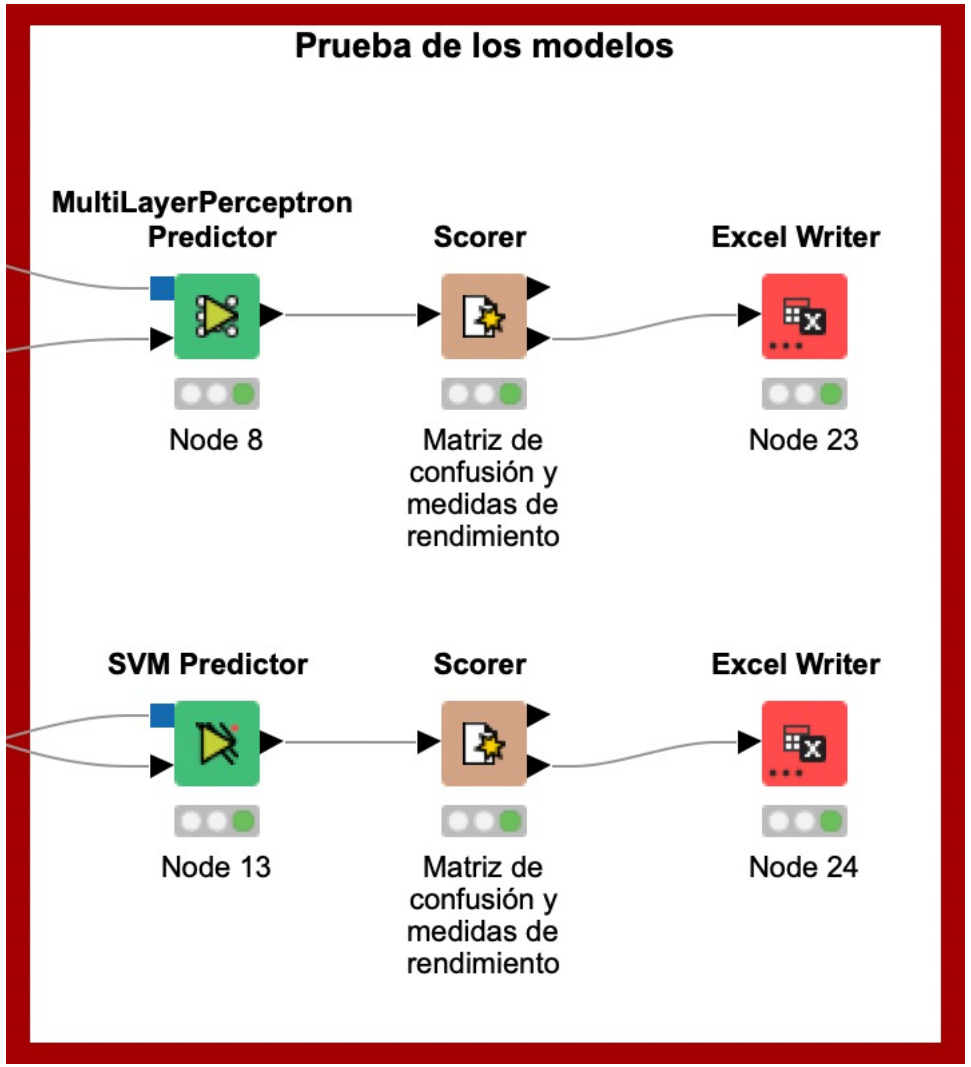
Knime: ionosphere

2. Entrenamiento



Knime: ionosphere

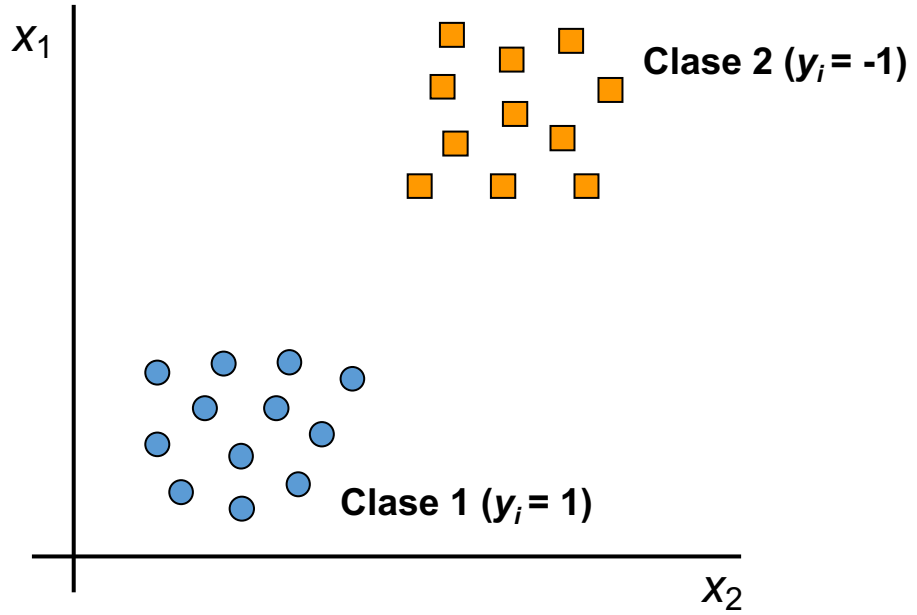
3. Prueba



SVM lineal

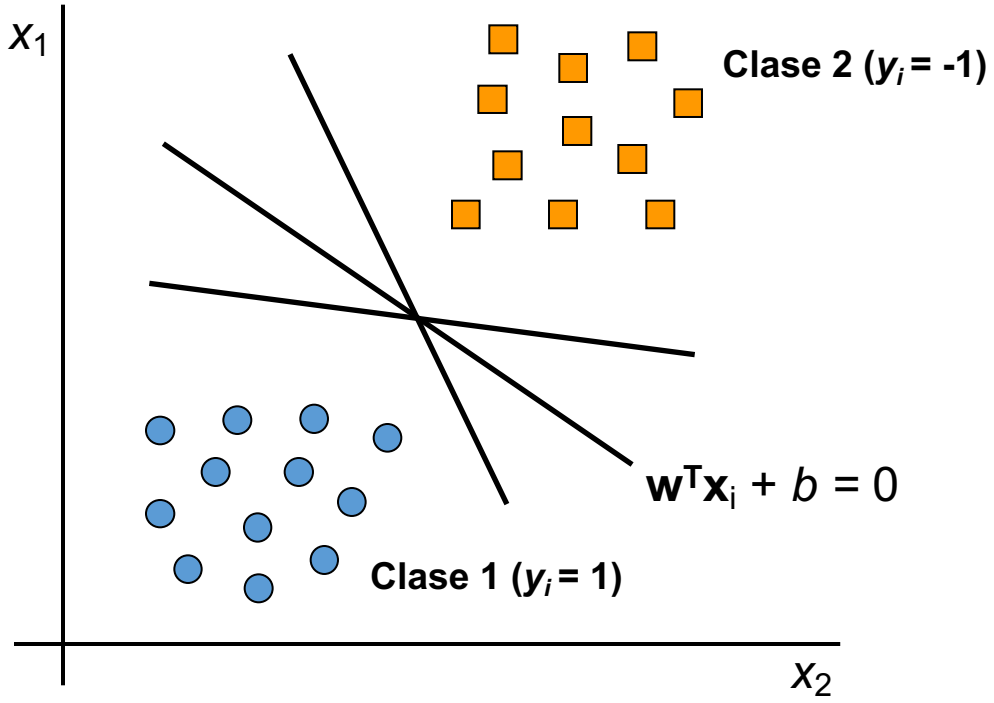
- Dado un conjunto de *N* datos (datos de entrenamiento) dividido en *dos* clases:

$$\{(\mathbf{x}_i, y_i)\},\ i = 1, \ldots, N$$
$$\mathbf{x}_i \in R^d$$
$$y_i \in \{-1, +1\}$$



- **Objetivo**: Desarrollar un método que sea capaz de discriminar *linealmente* entre ambas clases

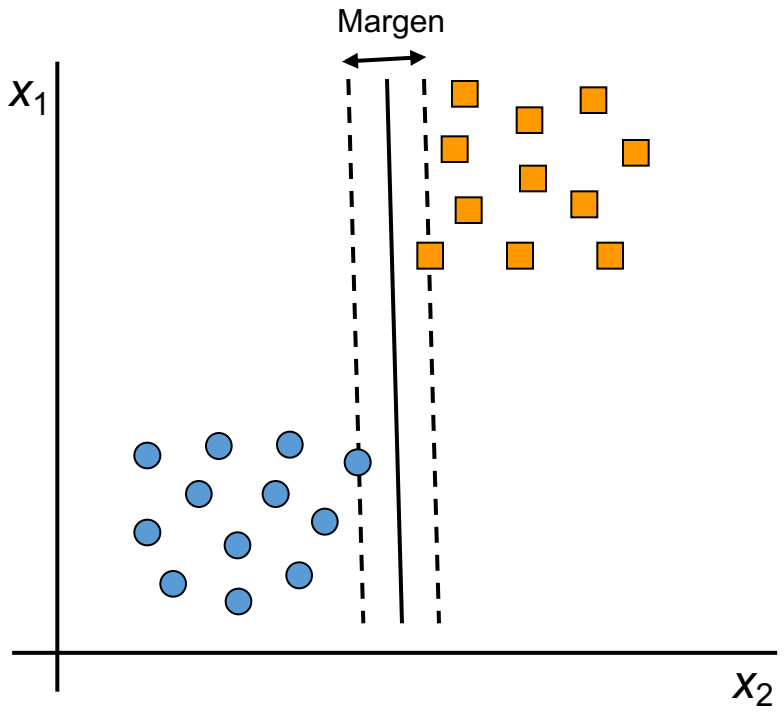
SVM lineal



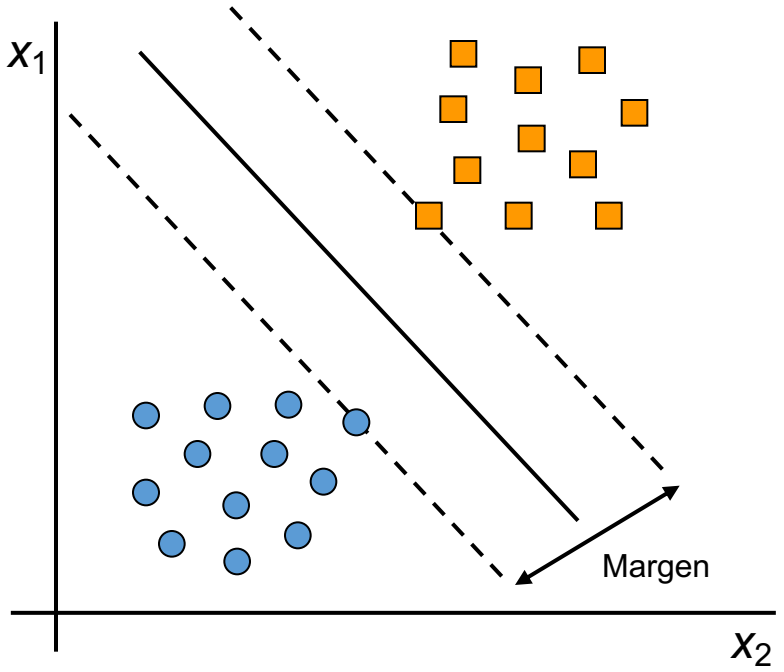
Pueden existir diversos hiperplanos óptimos en términos de minimización del error empírico (entrenamiento)

¿Son todos iguales en términos del error de generalización?

SVM lineal



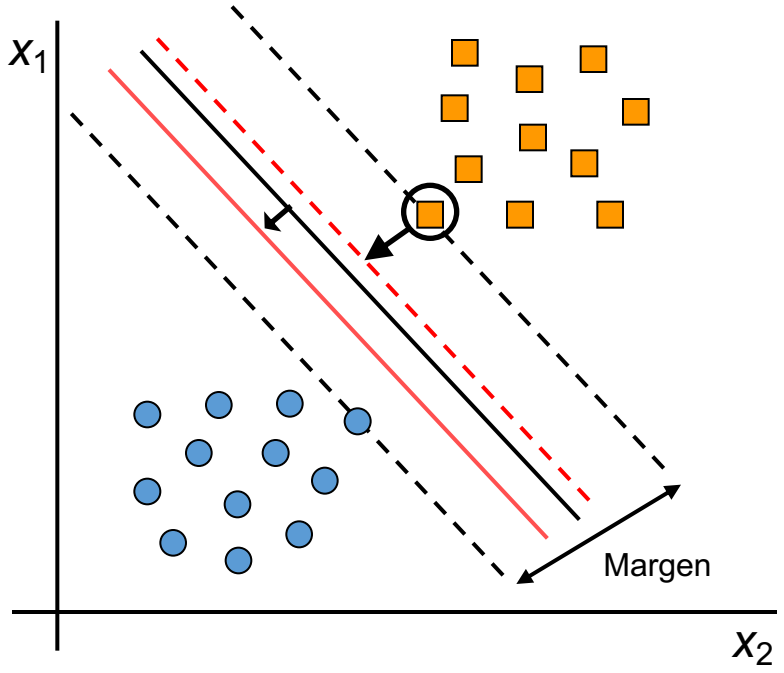
Menor margen:
Probablemente peor generalización



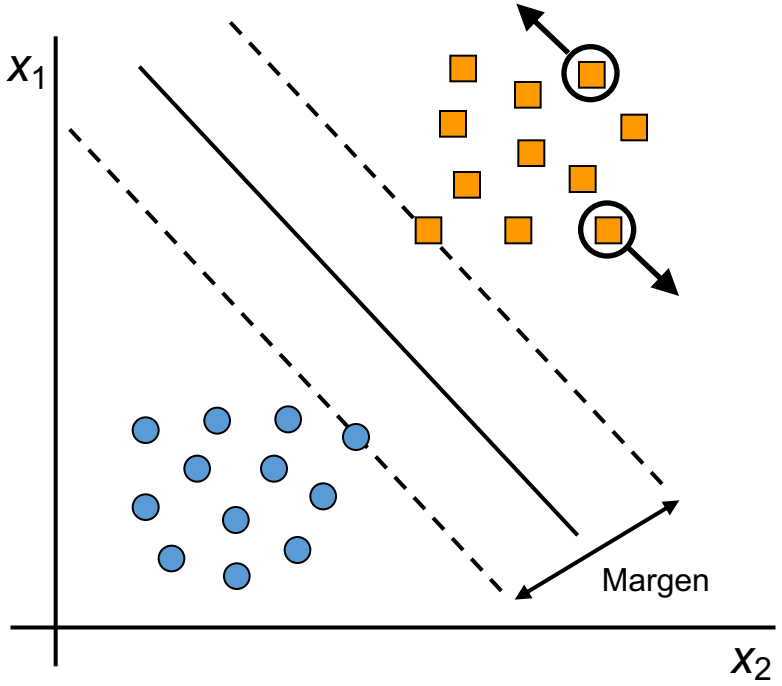
Mayor margen:
Probablemente buena generalización

Entre todos los posibles hiperplanos que minimizan el error empírico (entrenamiento), se debe elegir aquel con el mayor margen (Vapnik 1992)

SVM lineal



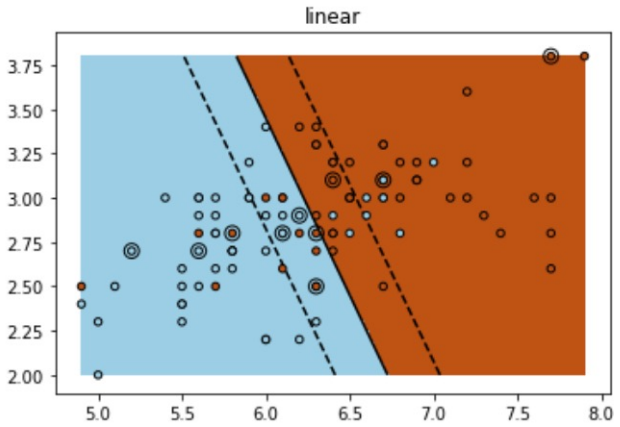
Si se modifica un dato que es un vector soporte, la región de decisión cambia



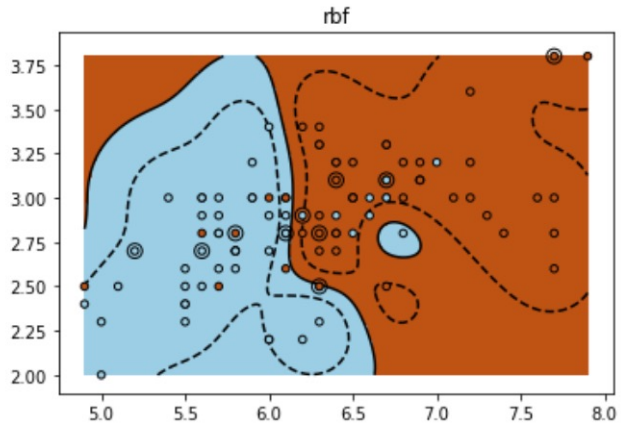
Si se modifica cualquier otro dato, la región de decisión no cambia (excepto si se convierte en un vector soporte)

SVM no lineal: kernels

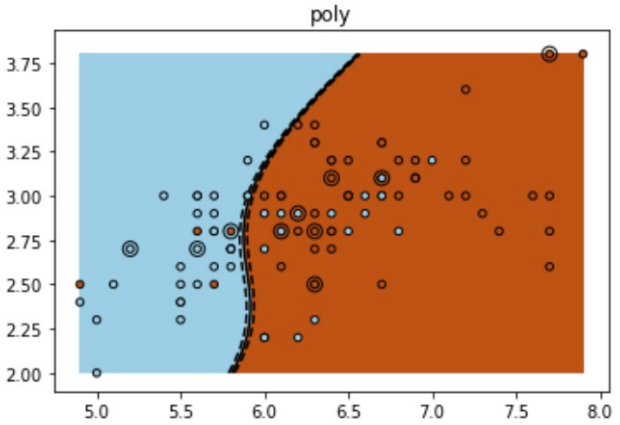
Lineal C= 1



Gaussiano C= 1,γ= 10



Polinómico C= 1,*orden*= 10



Ejercicio propuesto: Iris

- **Resolver un problema de clasificación mediante el uso de una red neuronal de tipo perceptrón multicapa.**
- El problema consiste en clasificar tres tipos de lirio a partir de atributos relacionados con la geometría de la planta
- Las muestras presentan 4 atributos de entrada y 1 de salida:
  - Longitud del sépalo (cm)
  - Ancho del sépalo (cm)
  - Longitud del pétalo (cm)
  - Ancho del pétalo (cm)

- Clase de lirio (1: Iris Setosa; 2: Iris Versicolor; 3: Iris Virginica)
- El conjunto de datos y una descripción de los mismos están disponibles en:
  - https://archive.ics.uci.edu/ml/datasets/Iris